KhaDev PRESENT

WORKSHOP ADVANCED DOCKER

WITH

kubernetes

สอนการ deploy Dockers บน Kubernetes
จากประสบการณ์ใช้งานจริงบน Production ของ
application ระดับประเทศ

วิทยากร : คุณ PRAPARN LUNGPOONLARP
INFRASTRUCTURE ENGINEER, NETWORK ENGINEER,
SYSTEM ENGINEER

docker kubernetes KhaDev 3DS INTERACTIVE

Outline Part 1

- Container concept (Recap)
- Introduction to Kubernetes
- System Architecture
- Fundamental of Kubernetes
  - Pods, Container and Services
  - Daemon Sets and Replication Controller (RC)
  - Deployment/Replica-Set (RS) and Rolling update
  - Volume
  - Resource Management and Horizontal Pods Autoscaling (HPA)
  - Liveness, Readiness and Startup Probe
  - ConfigMap Secret

Outline Part 2

- Fundamental of Kubernetes
  - Job and CronJob
  - Log and Monitoring
- Ingress & Gateway API Networking
- Security on Kubernetes
  - Network Policy
  - Volume Policy
  - Resource Usage Policy
  - Resource Consumption Policy
  - Access Control Policy
  - Security Policy
- Kubernetes in real world
  - Cluster Setup for Bare Metal
  - Orchestrator Assignment
    - nodeSelector
    - Interlude
    - Affinity
    - Taints/Tolerations
- Stateful application deployment
  - Consideration and Awareness
  - Persistent Volumes
  - StatefulSets
- Helm basic operation
- Monitoring with prometheus operator

Pre-require

- Windows 11 (64 bit)/ Mac OSX (64 Bit) with memory 8 GB
- Intermediate understand for docker/compose and container concept
- Cloud / Bare Material /Play with K8S
- LINE Account
- Basic understand for network/load balance concept
- Tool for editor (atom etc)
- Tool for shell (putty / terminal etc)
- Tool for transfer file (winscp / scp)
- Internet for download / upload image

LabSheet

- https://tinyurl.com/y8yazrxy

LabSheet

ไฟล์ แก้ไข ดู แทรก รูปแบบ ข้อมูล เครื่องมือ ส่วนเสริม ความช่วยเหลือ แก้ไขล่าสุด 1 นาทีที่ผ่านมา

Group No

Group No	No	Machine Name	Privage IP Address	Public IP Address	Role	Username	ssh connect string		Remark
	1	Training_AdvanceDockerwithK8S_StudentG1_1	10.0.1.109	13.229.92.115	Master	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.92.115	
	2	Training_AdvanceDockerwithK8S_StudentG1_2	10.0.1.139	13.250.11.169	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	13.250.11.169	
1	3	Training_AdvanceDockerwithK8S_StudentG1_3	10.0.1.47	52.221.249.56	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	52.221.249.56	
	4	Training_AdvanceDockerwithK8S_StudentG2_1	10.0.1.50	13.229.215.190	Master	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.215.190	
	5	Training_AdvanceDockerwithK8S_StudentG2_2	10.0.1.196	13.229.72.95	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.72.95	
2	6	Training_AdvanceDockerwithK8S_StudentG2_3	10.0.1.77	13.229.251.109	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.251.109	
	7	Training_AdvanceDockerwithK8S_StudentG3_1	10.0.1.174	54.169.75.239	Master	ubuntu	ssh -i "docker_lab" ubuntu@	54.169.75.239	
	8	Training_AdvanceDockerwithK8S_StudentG3_2	10.0.1.252	13.229.247.143	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.247.143	
3	9	Training_AdvanceDockerwithK8S_StudentG3_3	10.0.1.75	13.229.65.47	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.65.47	
	10	Training_AdvanceDockerwithK8S_StudentG4_1	10.0.1.85	13.229.206.76	Master	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.206.76	
	11	Training_AdvanceDockerwithK8S_StudentG4_2	10.0.1.208	13.229.70.154	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	13.229.70.154	
4	12	Training_AdvanceDockerwithK8S_StudentG4_3	10.0.1.86	3.0.147.248	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.147.248	
	13	Training_AdvanceDockerwithK8S_StudentG5_1	10.0.1.234	3.0.146.98	Master	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.146.98	
	14	Training_AdvanceDockerwithK8S_StudentG5_2	10.0.1.209	54.255.172.4	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	54.255.172.4	
5	15	Training_AdvanceDockerwithK8S_StudentG5_3	10.0.1.207	54.255.136.224	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	54.255.136.224	
	16	Training_AdvanceDockerwithK8S_StudentG6_1	10.0.1.134	54.169.204.141	Master	ubuntu	ssh -i "docker_lab" ubuntu@	54.169.204.141	
	17	Training_AdvanceDockerwithK8S_StudentG6_2	10.0.1.62	3.0.183.181	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.183.181	
6	18	Training_AdvanceDockerwithK8S_StudentG6_3	10.0.1.95	18.136.199.90	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	18.136.199.90	
	19	Training_AdvanceDockerwithK8S_StudentG7_1	10.0.1.202	3.0.176.230	Master	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.176.230	
	20	Training_AdvanceDockerwithK8S_StudentG7_2	10.0.1.41	3.0.94.126	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.94.126	
7	21	Training_AdvanceDockerwithK8S_StudentG7_3	10.0.1.81	3.0.54.130	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.54.130	
	22	Training_AdvanceDockerwithK8S_StudentG8_1	10.0.1.65	3.0.98.215	Master	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.98.215	
	23	Training_AdvanceDockerwithK8S_StudentG8_2	10.0.1.93	18.136.210.250	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	18.136.210.250	
8	24	Training_AdvanceDockerwithK8S_StudentG8_3	10.0.1.146	54.254.214.135	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	54.254.214.135	
	25	Training_AdvanceDockerwithK8S_StudentG9_1	10.0.1.154	54.255.138.111	Master	ubuntu	ssh -i "docker_lab" ubuntu@	54.255.138.111	
	26	Training_AdvanceDockerwithK8S_StudentG9_2	10.0.1.223	3.0.92.216	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.92.216	
9	27	Training_AdvanceDockerwithK8S_StudentG9_3	10.0.1.224	3.0.17.138	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.17.138	
	28	Training_AdvanceDockerwithK8S_StudentG10_1	10.0.1.55	13.250.38.8	Master	ubuntu	ssh -i "docker_lab" ubuntu@	13.250.38.8	
	29	Training_AdvanceDockerwithK8S_StudentG10_2	10.0.1.149	52.221.184.17	Worker1	ubuntu	ssh -i "docker_lab" ubuntu@	52.221.184.17	
10	30	Training_AdvanceDockerwithK8S_StudentG10_3	10.0.1.17	3.0.50.103	Worker2	ubuntu	ssh -i "docker_lab" ubuntu@	3.0.50.103	

Present by: Praparn L. (eva10409@gmail.com)

kubernetes by Google

Lab Resource

- Repository for lab

Explore Help labdocker Sign up Log In

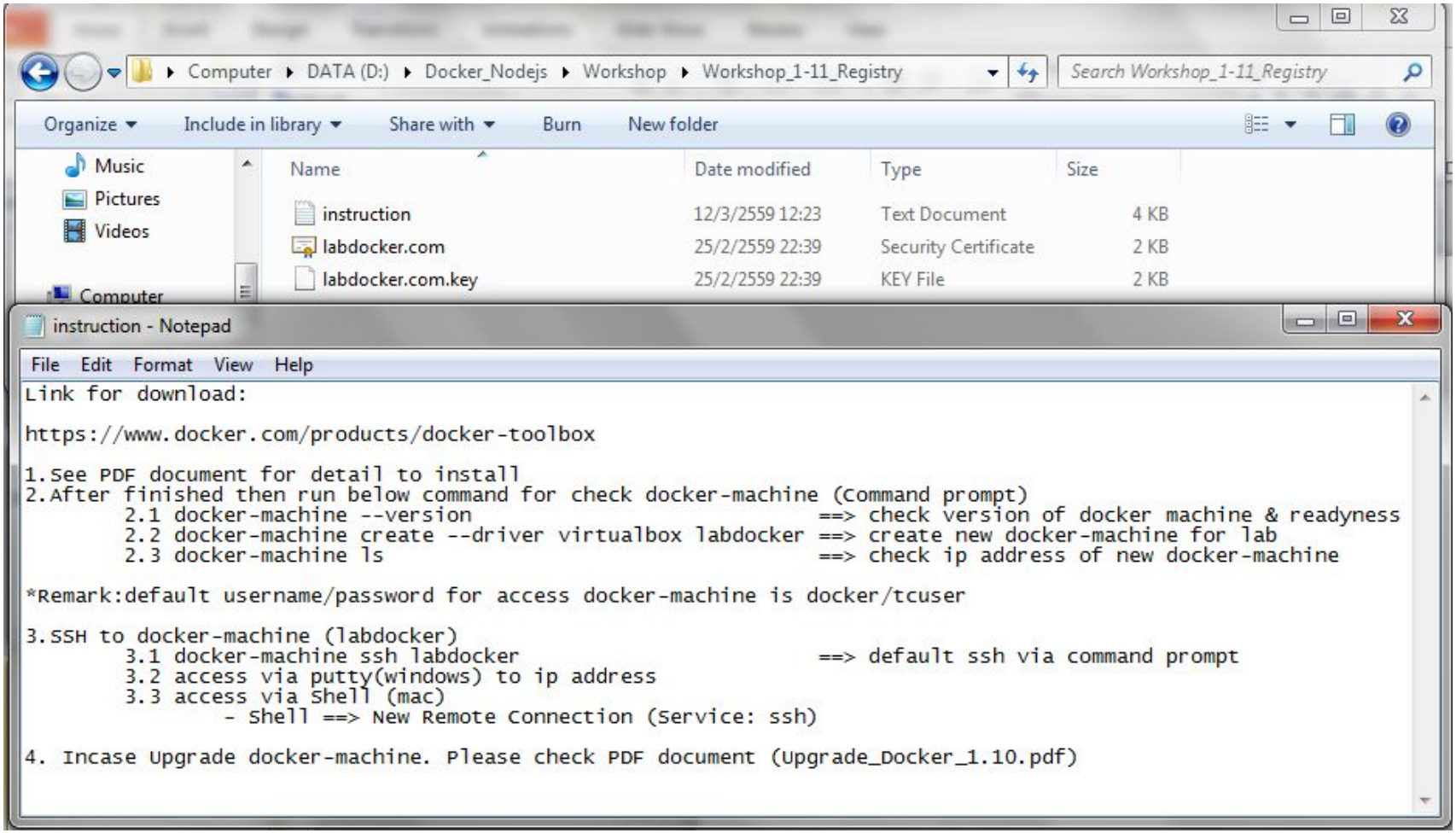
Repositories (7)

All

labdocker/alpineweb public	0 STARS	31 PULLS	DETAILS
labdocker/nginx public	0 STARS	16 PULLS	DETAILS
labdocker/alpine public	0 STARS	7 PULLS	DETAILS

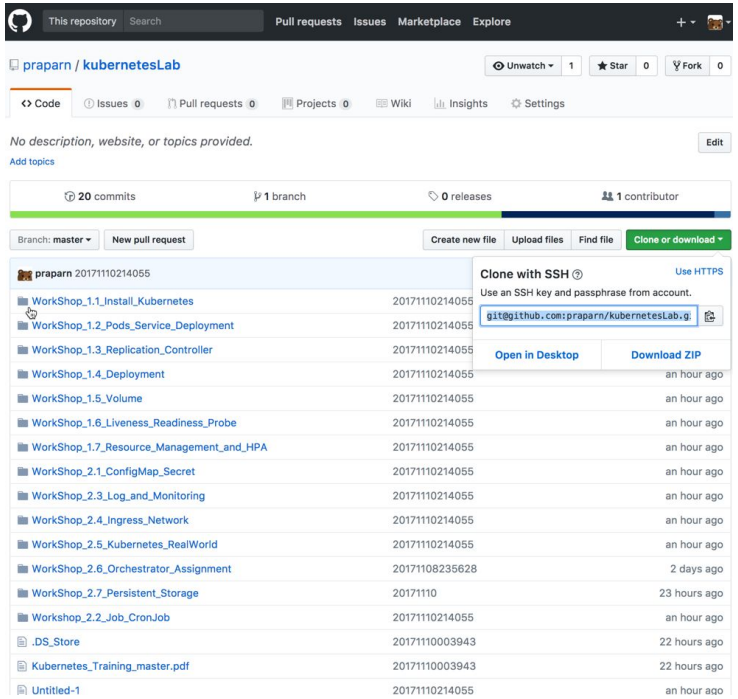
Lab Resource

- Software in lab



Lab Resource

- Download on Google Drive
  - https://tinyurl.com/k8slabth
- Download on GitHub
  - Git clone https://github.com/praparn/kubernetes_202405



Setup: https://github.com/praparn/sourcesetup/blob/master/standard_kubernetes300_containerd.sh

Workshop: Install Kubenetes

Workshop: Install Kubenetes

- LAB Sheet
- https://tinyurl.com/y8yazrxy

Table No	Group No	No	Public IP Address	Privage IP Address	Role
		1	54.169.12.33	10.0.1.25	Master
		2	54.255.243.139	10.0.1.102	Worker
	1	3	52.221.203.26	10.0.1.254	Worker
		1	13.250.121.106	10.0.1.100	Master
		2	54.255.243.173	10.0.1.202	Worker
1	2	3	52.221.232.245	10.0.1.12	Worker
		1	54.169.39.94	10.0.1.169	Master
		2	13.229.72.232	10.0.1.107	Worker
	3	3	13.250.109.203	10.0.1.135	Worker
		1	13.250.103.188	10.0.1.110	Master
		2	13.229.128.105	10.0.1.99	Worker
2	4	3	54.169.134.39	10.0.1.162	Worker
		1	54.255.145.5	10.0.1.32	Master
		2	54.254.218.192	10.0.1.59	Worker
	5	3	52.221.216.253	10.0.1.166	Worker
		1	13.250.112.82	10.0.1.45	Master
		2	54.254.166.86	10.0.1.50	Worker
3	6	3	52.221.181.188	10.0.1.218	Worker
		1	52.221.208.50	10.0.1.182	Master
		2	52.77.249.187	10.0.1.4	Worker
	7	3	13.250.104.158	10.0.1.173	Worker
		1	54.169.32.99	10.0.1.222	Master
		2	52.221.191.74	10.0.1.147	Worker
4	8	3	13.229.80.170	10.0.1.167	Worker

Kubernetes 1.30 Uwubernetes

Graduated to stable

This lists all the features that graduated to stable (also known as *general availability*). For a full list of updates including new features and graduations from alpha to beta, see the release notes.

This release includes a total of 17 enhancements promoted to Stable:

- Container Resource based Pod Autoscaling
- Remove transient node predicates from KCCM's service controller
- Go workspaces for k/k
- Reduction of Secret-based Service Account Tokens
- CEL for Admission Control
- CEL-based admission webhook match conditions
- Pod Scheduling Readiness
- Min domains in PodTopologySpread
- Prevent unauthorised volume mode conversion during volume restore
- API Server Tracing
- Cloud Dual-Stack --node-ip Handling
- AppArmor support
- Robust VolumeManager reconstruction after kubelet restart
- kubectl delete: Add interactive(-i) flag
- Metric cardinality enforcement
- Field `status.hostIPs` added for Pod
- Aggregated Discovery

Deprecations and removals

Removed the SecurityContextDeny admission plugin, deprecated since v1.27

(SIG Auth, SIG Security, and SIG Testing) With the removal of the SecurityContextDeny admission plugin, the Pod Security Admission plugin, available since v1.25, is recommended instead.

Ref: https://kubernetes.io/blog/2024/04/17/kubernetes-v1-30-release

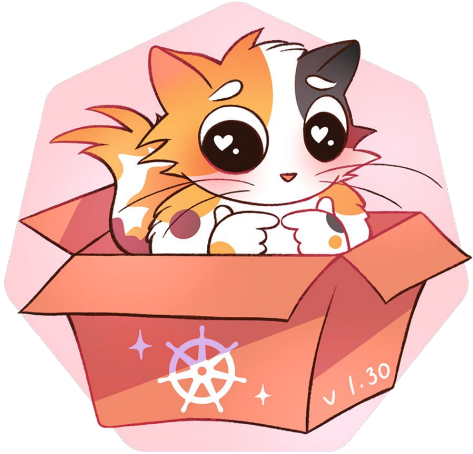
Kubernetes 1.30 Uwubernetes

Kubernetes v1.30 makes your clusters cuter!

Kubernetes is built and released by thousands of people from all over the world and all walks of life. Most contributors are not being paid to do this; we build it for fun, to solve a problem, to learn something, or for the simple love of the community. Many of us found our homes, our friends, and our careers here. The Release Team is honored to be a part of the continued growth of Kubernetes.

For the people who built it, for the people who release it, and for the furries who keep all of our clusters online, we present to you Kubernetes v1.30: Uwubernetes, the cutest release to date. The name is a portmanteau of "kubernetes" and "UwU," an emoticon used to indicate happiness or cuteness. We've found joy here, but we've also brought joy from our outside lives that helps to make this community as weird and wonderful and welcoming as it is. We're so happy to share our work with you.

UwU ♥



Robust VolumeManager reconstruction after kubelet restart (SIG Storage)

This is a volume manager refactoring that allows the kubelet to populate additional information about how existing volumes are mounted during the kubelet startup. In general, this makes volume cleanup after kubelet restart or machine reboot more robust.

This does not bring any changes for user or cluster administrators. We used the feature process and feature gate `NewVolumeManagerReconstruction` to be able to fall back to the previous behavior in case something goes wrong. Now that the feature is stable, the feature gate is locked and cannot be disabled.

Prevent unauthorized volume mode conversion during volume restore (SIG Storage)

For Kubernetes v1.30, the control plane always prevents unauthorized changes to volume modes when restoring a snapshot into a PersistentVolume. As a cluster administrator, you'll need to grant permissions to the appropriate identity principals (for example: ServiceAccounts representing a storage integration) if you need to allow that kind of change at restore time.

> **Warning:** Action required before upgrading. The `prevent-volume-mode-conversion` feature flag is enabled by default in the external-provisioner `v4.0.0` and external-snapshotter `v7.0.0`. Volume mode change will be rejected when creating a PVC from a VolumeSnapshot unless you perform the steps described in the the "Urgent Upgrade Notes" sections for the external-provisioner 4.0.0 and the external-snapshotter v7.0.0.

Ref: https://kubernetes.io/blog/2024/04/17/kubernetes-v1-30-release

Kubernetes 1.30 Uwubernetes

Pod Scheduling Readiness (SIG Scheduling)

Pod scheduling readiness graduates to stable this release, after being promoted to beta in Kubernetes v1.27.

This now-stable feature lets Kubernetes avoid trying to schedule a Pod that has been defined, when the cluster doesn't yet have the resources provisioned to allow actually binding that Pod to a node. That's not the only use case; the custom control on whether a Pod can be allowed to schedule also lets you implement quota mechanisms, security controls, and more.

Crucially, marking these Pods as exempt from scheduling cuts the work that the scheduler would otherwise do, churning through Pods that can't or won't schedule onto the nodes your cluster currently has. If you have cluster autoscaling active, using scheduling gates doesn't just cut the load on the scheduler, it can also save money. Without scheduling gates, the autoscaler might otherwise launch a node that doesn't need to be started.

In Kubernetes v1.30, by specifying (or removing) a Pod's `.spec.schedulingGates`, you can control when a Pod is ready to be considered for scheduling. This is a stable feature and is now formally part of the Kubernetes API definition for Pod.

Min domains in PodTopologySpread (SIG Scheduling)

The `minDomains` parameter for PodTopologySpread constraints graduates to stable this release, which allows you to define the minimum number of domains. This feature is designed to be used with Cluster Autoscaler.

If you previously attempted use and there weren't enough domains already present, Pods would be marked as unschedulable. The Cluster Autoscaler would then provision node(s) in new domain(s), and you'd eventually get Pods spreading over enough domains.

Go workspaces for k/k (SIG Architecture)

The Kubernetes repo now uses Go workspaces. This should not impact end users at all, but does have a impact for developers of downstream projects. Switching to workspaces caused some breaking changes in the flags to the various k8s.io/code-generator tools. Downstream consumers should look at `staging/src/k8s.io/code-generator/kube_codegen.sh` to see the changes.

For full details on the changes and reasons why Go workspaces was introduced, read Using Go workspaces in Kubernetes.

Ref: https://kubernetes.io/blog/2024/04/17/kubernetes-v1-30-release

Kubernetes 1.30 Uwubernetes

Preview Code Blame 1085 lines (839 loc) · 44.3 KB Raw

KEP-1610: Container Resource based Autoscaling

Summary

The Horizontal Pod Autoscaler supports scaling of targets based on the resource usage of the pods in the target. The resource usage of pods is calculated as the sum of the individual container usage values of the pod. This is unsuitable for workloads where the usage of the containers are not strongly correlated or do not change in lockstep. This KEP suggests that when scaling based on resource usage the HPA also provide an option to consider the usages of individual containers to make scaling decisions.

Motivation

An HPA is used to ensure that a scaling target is scaled up or down in such a way that the specificed current metric values are always maintained at a certain level. Resource based autoscaling is the most basic approach to autoscaling and has been present in the HPA spec since `v1`. In this mode the HPA controller fetches the current resource metrics for all the pods of a scaling target and then computes how many pods should be added or removed based on the current usage to achieve the target average usage.

For performance critical applications where the resource usage of individual containers needs to be configured individually the default behavior of the HPA controller may be unsuitable. When there are multiple containers in the pod their individual resource usages may not have a direct correlation or may grow at different rates as the load changes. There are several reasons for this:

- A sidecar container is only providing an auxiliary service such as log shipping. If the application does not log very frequently or does not produce logs in its hotpath then the usage of the log shipper will not grow.
- A sidecar container which provides authentication. Due to heavy caching the usage will only increase slightly when the load on the main container increases. In the current blended usage calculation approach this usually results in the the HPA not scaling up the deployment because the blended usage is still low.
- A sidecar may be injected without resources set which prevents scaling based on utilization. In the current logic the HPA controller can only scale on absolute resource usage of the pod when the resource requests are not set.

The optimum usage of the containers may also be at different levels. Hence the HPA should offer a way to specify the target usage in a more fine grained manner.

Ref: https://github.com/kubernetes/enhancements/blob/master/keps/sig-autoscaling/1610-container-resource-autoscaling/README.md#summary

Kubernetes 1.30 Uwubernetes

Preview | Code | Blame | 1085 lines (839 loc) · 44.3 KB | Raw

KEP-1610: Container Resource based Autoscaling

```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: mission-critical
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: mission-critical
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: ContainerResource
    resource:
      name: cpu
      container: application
      target:
        type: Utilization
        averageUtilization: 50
  - type: PodResource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
```

```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: mission-critical
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: mission-critical
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: ContainerResource
    resource:
      name: cpu
      container: application
      target:
        type: Utilization
        averageUtilization: 30
  - type: ContainerResource
    resource:
      name: memory
      container: application
      target:
        type: Utilization
        averageUtilization: 80
```

```yaml
  - type: ContainerResource
    resource:
      name: cpu
      container: authnz-proxy
      target:
        type: Utilization
        averageUtilization: 30
  - type: ContainerResource
    resource:
      name: memory
      container: authnz-proxy
      target:
        type: Utilization
        averageUtilization: 80
  - type: ContainerResource
    resource:
      name: cpu
      container: log-shipping
      target:
        type: Utilization
        averageUtilization: 80
```


kubernetes by Google

Kubernetes 1.30 Uwubernetes

Preview Code Blame 933 lines (729 loc) · 36 KB Raw

KEP-3895: Interactive(-i) flag to kubectl delete for user confirmation

Summary

Introduce interactive mode for `kubectl delete` command, to allow users protect cluster administrators from irrevocable removal of critical resources.

Motivation

kubectl delete is disruptive and irreversible command and should be used cautiously. However, there are various circumstances such as inaccurate usage, mistyping, hasty tab completions, etc. that this command can position the cluster to a state where it is not intended to be. Due to the backwards compatibility concerns, seeking confirmation from a user before proceeding to genuinely delete as default, is out of the table.

On the other hand, we need to provide a way to mitigate accidental deletes to users and this KEP proposes a new flag, namely interactive(-i) that will ask confirmation from a user to proceed with deletion.

Proposal

When a user passes `--interactive` / `-i` flag to `kubectl delete` command, all the resources planned to be deleted will be shown to the user as preview. Command continues deletion process if user confirms by typing `y`. If user types `n` or any other value other than `y`, process will be cancelled and command returns a message with zero exit code.

```
$ kubectl delete --interactive -f test_deployment.yaml
You are about to delete the following 2 resource(s):
Deployment/dockerd
Deployment/dockerd2
Do you want to continue? (y/n): n
deletion is cancelled

$ kubectl delete -i -f test_deployment.yaml
You are about to delete the following 2 resource(s):
Deployment/dockerd
Deployment/dockerd2
Do you want to continue? (y/n): y
deployment.apps "dockerd" deleted
deployment.apps "dockerd2" deleted
```

Kubernetes 1.30 Uwubernetes

README.md

KEP-3022: min domains in Pod Topology Spread

Summary

A new field `minDomains` is introduced to `PodSpec.TopologySpreadConstraint[*]` to limit the minimum number of topology domains. `minDomains` can be used only when `whenUnsatisfiable=DoNotSchedule`.

Motivation

Pod Topology Spread has `maxSkew` parameter, which control the degree to which Pods may be unevenly distributed. But, there isn't a way to control the number of domains over which we should spread. In some cases, users want to force spreading Pods over a minimum number of domains and, if there aren't enough already present, make the cluster-autoscaler provision them.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 10
  template:
    metadata:
      labels:
        foo: bar
    spec:
      containers:
        - name: nginx
          image: nginx:1.14.2
          ports:
            - containerPort: 80
      topologySpreadConstraints:
        - maxSkew: 2
          minDomains: 5
          topologyKey: kubernetes.io/hostname
          whenUnsatisfiable: DoNotSchedule
          labelSelector:
            matchLabels:
              foo: bar
```

Kubernetes 1.30 Uwubernetes

README.md

KEP-2681: Field status.hostIPs added for Pod

Summary

The proposal aims to improve the Pod's ability to obtain the address of the node

Motivation

Goals

- Field `status.hostIPs` added for Pod
- Downward API support for `status.hostIPs`

```yaml
- name: MY_HOST_IPS
  valueFrom:
    fieldRef:
      fieldPath: status.hostIPs
```

```
$ kubectl get pod
NAME                                    READY   STATUS    RESTARTS   AGE
agnhost-server-76fb5c696c-2rqnh         1/1     Running   0          6s
```

Check pod hostIPs

```
$ kubectl get pod agnhost-server-76fb5c696c-2rqnh -o jsonpath='{.status.hostIPs}'
[{"ip":"172.18.0.2"}]
```

Kubernetes 1.30 Uwubernetes

README.md

KEP-2799: Reduction of Secret-based Service Account Tokens

Summary

This KEP proposes actions to reduce the surface area of secret-based service account tokens.

Motivation

As BoundServiceAccountTokenVolume is GA in 1.22, pods' service account tokens would be obtained via TokenRequest API and stored as projected volume. This change obviates the need for auto-generation of secret-based service account tokens which are less secure than the bound token.

LegacyServiceAccountTokenCleanUp

Token Controller starts to remove unused auto-generated secrets (secrets bi-directionally referenced by the service account) and not mounted by pods.

When this feature is Beta and enabled by default, mark the secrets as invalid iff it is over a sufficient period of time (one year by default) since last used. The period can be configured by cluster admins.

Determine the date that a given secret was last used:

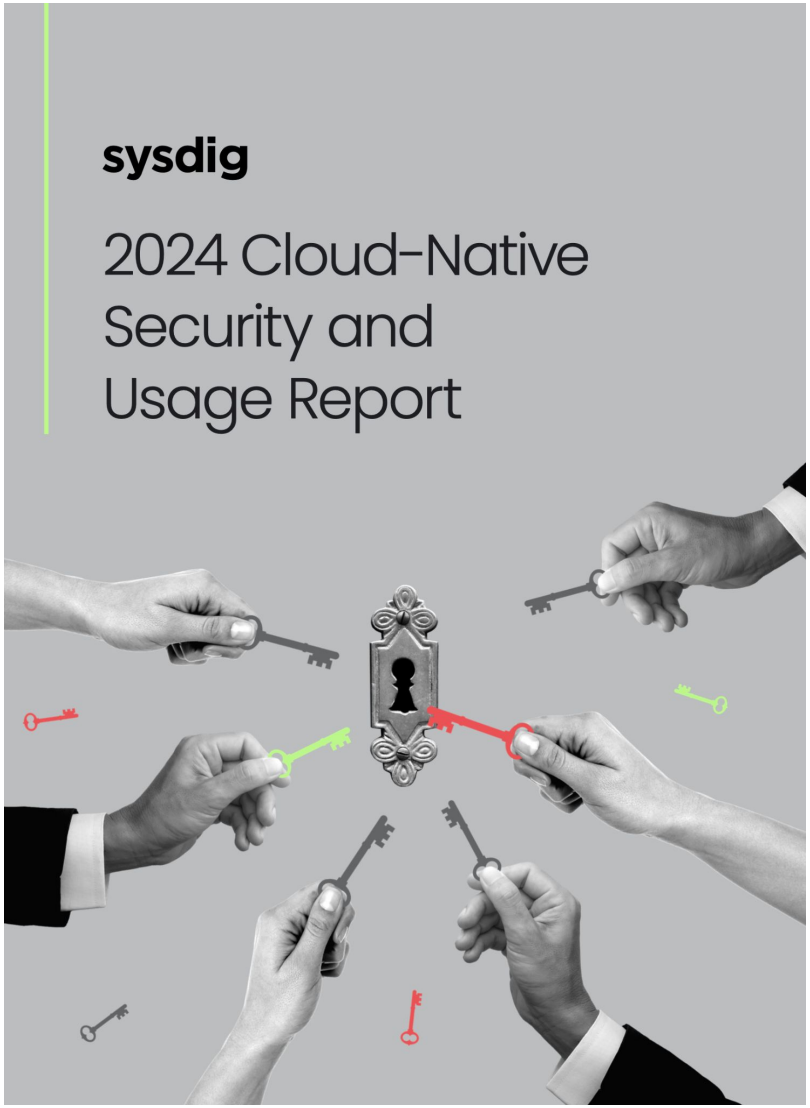
1. `kubernetes.io/legacy-token-last-used` if exists and after `since` stored in the configmap `kube-apiserver-legacy-service-account-token-tracking`.
2. defaults to `since`

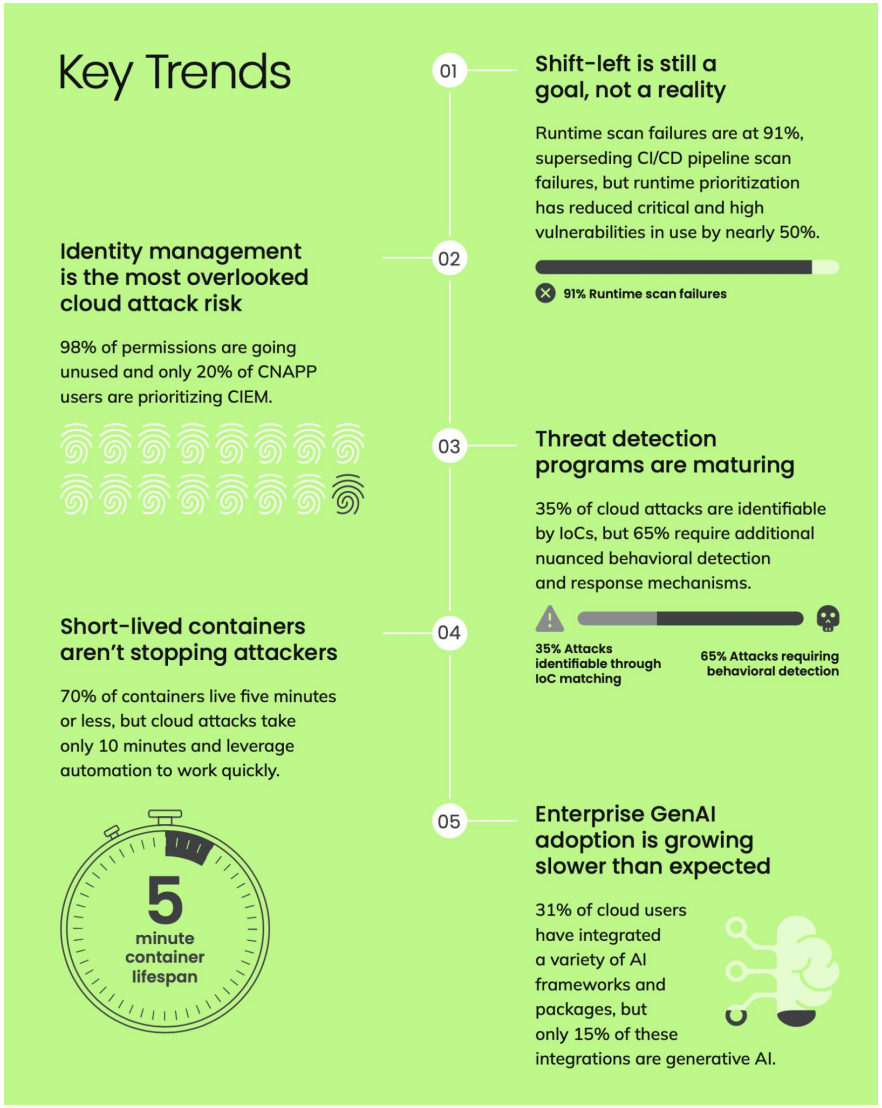
If `kube-apiserver-legacy-service-account-token-tracking` is unavailable, no secret would be removed.

Mark the secrets as invalid and recover:

1. The secrets will be added a label `kubernetes.io/legacy-token-invalid-since`, with the date as value.
2. If the users use the invalid tokens, in the Validate() function of "kubernetes/pkg/serviceaccount/legacy.go", it will detect the usage of invalid tokens and return the error information, telling the users to re-activate the token by updating the label value or use the tokenrequest. At the same time, the tokens will be updated with the new `kubernetes.io/legacy-token-last-used` date.
3. If the users don't use the invalid tokens, after the duration configured through `--legacy-service-account-token-clean-up-period` (one year by default) since the tokens are marked as invalid, the tokens will be finally deleted.

Sysdig Cloud Native 2024





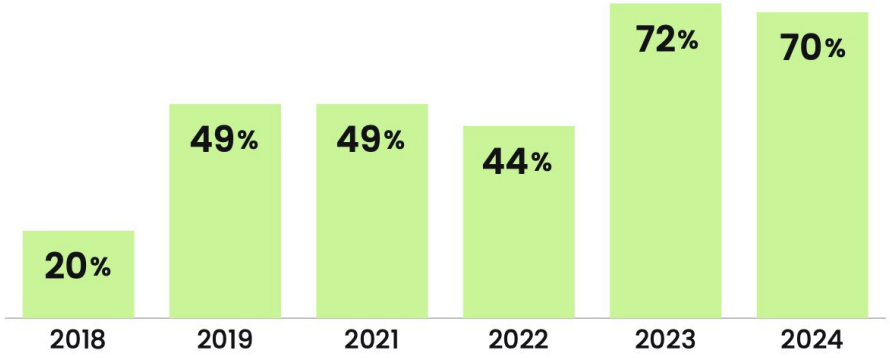
Kubernetes: Production Workload Orchestration

Sysdig Cloud Native 2024

Attackers have container lifespans beat

Year-over-year we have seen the average lifespan of a container get shorter. This year, 70% of containers are short-lived and spun down in five minutes or less. The Sysdig Threat Research Team (TRT) reported in the 2023 Global Cloud Threat Report that a cloud attack takes only 10 minutes. If an attacker has not moved laterally, they are booted once the container has been executed and killed. However, we know that it is fast and easy for an attacker to enter and move through an environment because, in the cloud, attackers automate their discovery and reconnaissance efforts. Almost in an instant, once an attacker is in an environment, they have the lay of the land and are ready to press forward. This highlights the importance of real-time security and continuous scanning. Running vulnerable workloads, no matter how short-lived, leaves an organization at risk for an attack.

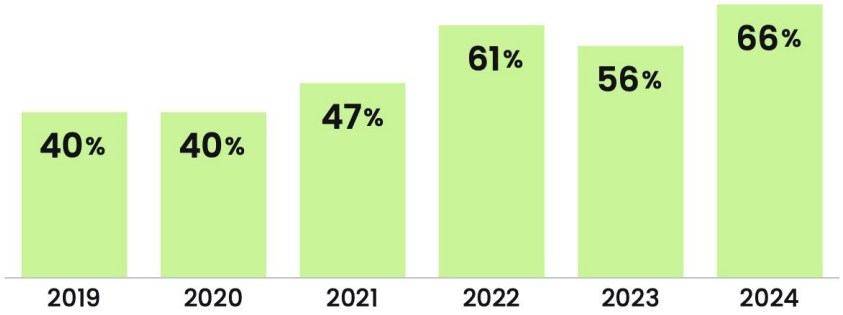
Containers living less than 5 minutes



Registry type

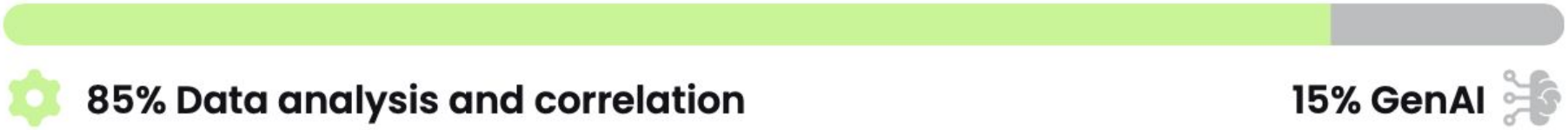
It's safe to say that after reviewing this data for several years, the convenience of public and managed registries is appealing to the majority of container users, regardless of their security maturity. Using public registries for images allows organizations to save time creating and maintaining their own registries and save money by not having to pay for vendor-managed registries. However, it is important to note that public registries come with reduced security control enforcement and create software supply chain risk.

Public registry use

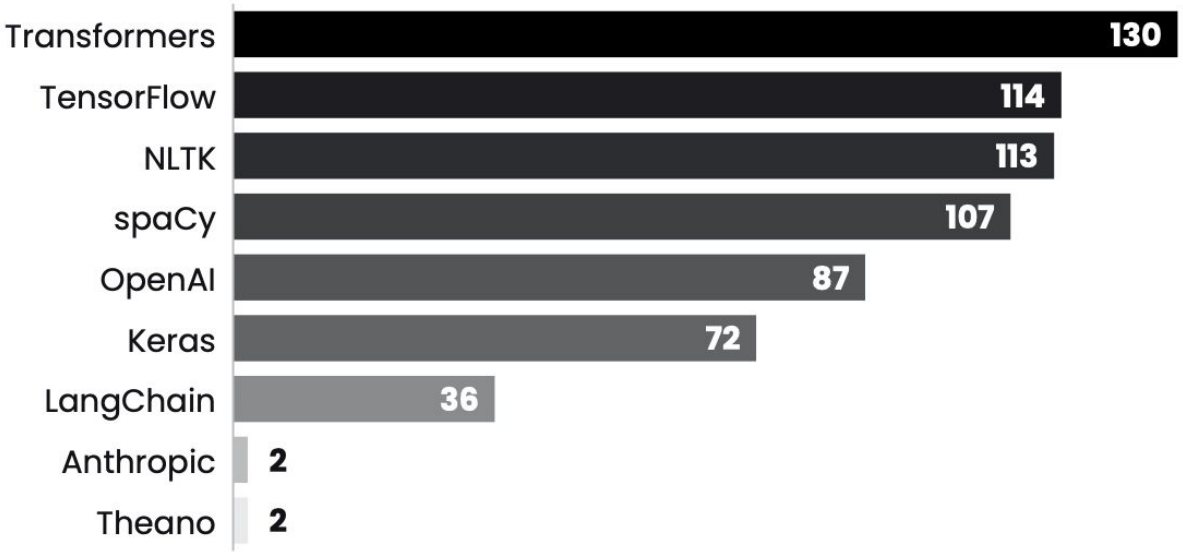


Sysdig Cloud Native 2024

AI adoption



GenAI package types



This table represents the 15% of GenAI package types seen in cloud environments

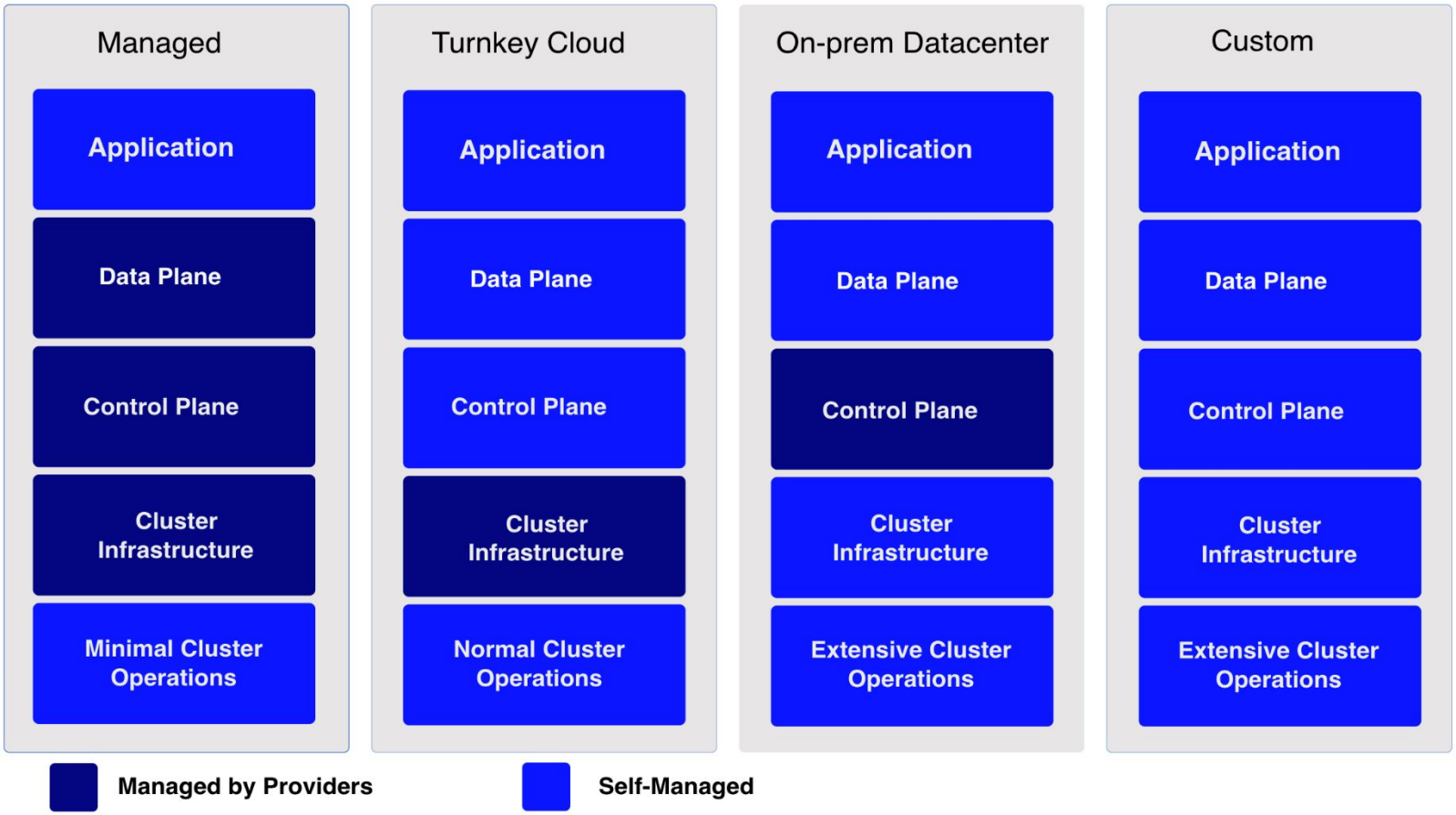
Kubernetes Environment

Production environment

When evaluating a solution for a production environment, consider which aspects of operating a Kubernetes cluster (or *abstractions*) you want to manage yourself or offload to a provider.

Some possible abstractions of a Kubernetes cluster are applications, data plane, control plane, cluster infrastructure, and cluster operations.

The following diagram lists the possible abstractions of a Kubernetes cluster and whether an abstraction is self-managed or managed by a provider.



Production environment solutions

Ref:https://kubernetes.io/docs/setup/release/notes/

Kubernetes: Production Workload Orchestration

Kubernetes Environment

Providers	Managed	Turnkey cloud	On-prem datacenter	Custom (cloud)	Custom (On-premises VMs)	Custom (Bare Metal)
Agile Stacks		✔	✔			
Alibaba Cloud		✔				
Amazon	Amazon EKS	Amazon EC2				
AppsCode	✔					
APPUiO	✔	✔	✔			
Banzai Cloud Pipeline Kubernetes Engine (PKE)		✔		✔	✔	✔
CenturyLink Cloud		✔				
Cisco Container Platform			✔			
Cloud Foundry Container Runtime (CFCR)				✔	✔	
CloudStack					✔	
Canonical	✔	✔	✔	✔	✔	✔
Containership	✔	✔				
Digital Rebar						✔
DigitalOcean	✔					
Docker Enterprise		✔	✔			✔
Fedora (Multi Node)					✔	✔
Fedora (Single Node)						✔
Gardener	✔	✔	✔ (via OpenStack)	✔		
Giant Swarm	✔	✔	✔			
Google	Google Kubernetes Engine (GKE)	Google Compute Engine (GCE)	GKE On-Prem			
IBM	IBM Cloud Kubernetes Service		IBM Cloud Private			
Ionos	Ionos Managed Kubernetes	Ionos Enterprise Cloud				
Kontena Pharos		✔	✔			
Kubermatic	✔	✔	✔			
KubeSail	✔					
Kubespray				✔	✔	✔
Kublr	✔	✔	✔	✔	✔	✔
Microsoft Azure	Azure Kubernetes Service (AKS)					
Mirantis Cloud Platform			✔			
Nirmata		✔	✔			
Nutanix	Nutanix Karbon	Nutanix Karbon			Nutanix AHV	
OpenShift	OpenShift Dedicated and OpenShift Online		OpenShift Container Platform		OpenShift Container Platform	OpenShift Container Platform

by Google

Kubernetes Environment

Oracle Cloud Infrastructure Container Engine for Kubernetes (OKE)	✔	✔				
oVirt					✔	
Pivotal		Enterprise Pivotal Container Service (PKS)	Enterprise Pivotal Container Service (PKS)			
Platform9	✔	✔	✔		✔	✔
Rancher		Rancher 2.x		Rancher Kubernetes Engine (RKE)		k3s
StackPoint	✔	✔				
Supergiant		✔				
SUSE		✔				
SysEleven	✔					
Tencent Cloud	Tencent Kubernetes Engine	✔	✔			✔
VEXXHOST	✔	✔				
VMware	VMware Cloud PKS	VMware Enterprise PKS	VMware Enterprise PKS	VMware Essential PKS		VMware Essential PKS
Z.A.R.V.I.S.	✔					

Introduction to Kubernetes

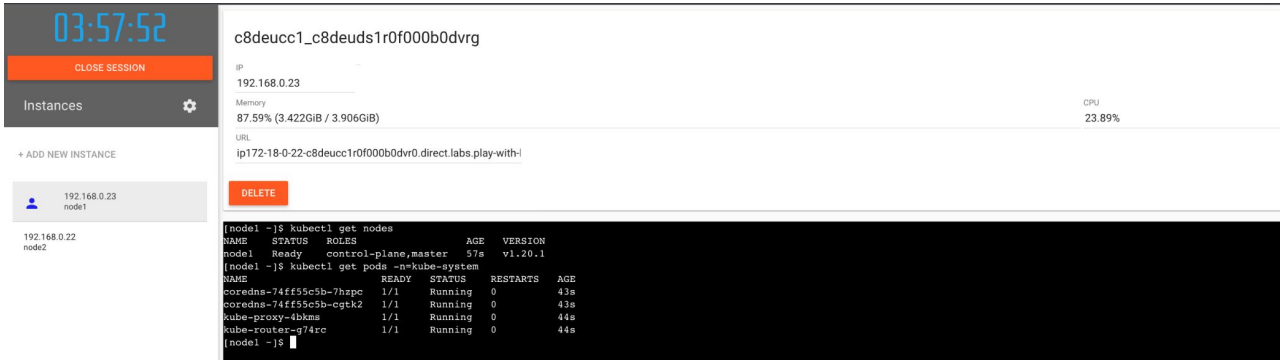
Kubernetes: Production Workload Orchestration

kubernetes by Google

Workshop: Install Kubernetes

Workshop : Install Kubernetes

- Alternate Solution:



Play with Kubernetes is a labs site provided by Docker and created by Tutorius. Play with Kubernetes is a playground which allows users to run K8s clusters in a matter of seconds. It gives the experience of having a free Alpine Linux Virtual Machine in browser. Under the hood Docker-in-Docker (DinD) is used to give the effect of multiple VMs/PCs.

If you want to learn more about Kubernetes, consider the Play with Kubernetes Classroom which provides more directed learning using an integrated Play with Kubernetes commandline.

Your use of Play With Docker is subject to the Docker Terms of Service which can be accessed here

Site provided by Docker, Inc., created by Tutorius

Introduction to Kubernetes

- Check kubernetes version

kubectl get nodes -o yaml

```
praparns-MacBook-Pro:~ praparn$ kubectl get nodes -o yaml
apiVersion: v1
items:
- apiVersion: v1
  kind: Node
  metadata:
    annotations:
      node.alpha.kubernetes.io/ttl: "0"
      volumes.kubernetes.io/controller-managed-attach-detach: "true"
    creationTimestamp: 2017-06-24T09:03:07Z
    labels:
      beta.kubernetes.io/arch: amd64
      beta.kubernetes.io/os: linux
      kubernetes.io/hostname: minikube
    name: minikube
    namespace: ""
    resourceVersion: "15676"
    selfLink: /api/v1/nodesminikube
    uid: f03de16d-58bb-11e7-aae1-080027559511
  spec:
    externalID: minikube
```

```
    nodeInfo:
      architecture: amd64
      bootID: cdf78a28-072d-4707-8c2d-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: ""
```

Introduction to Kubernetes

CLOUD NATIVE COMPUTING FOUNDATION

About Projects Certification People Community Newsroom JOIN NOW

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.

kubernetes Prometheus OPENTRACING fluentd GRPC rkt CNI envoy JAEGER TUF Vitess CoreDNS NATS LINKERD HELM ROOK

Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes

- What is Orchestration (Computing)?
  - Align business request with Application/Data/Infrastructure
  - Centralized management for:
    - Resource Pool
    - Automated Workflow
    - Provisioning
    - Scale Up/Down
    - Monitoring
    - Billing
    - etc

Ref: https://en.wikipedia.org/wiki/Orchestration_(computing)

Introduction to Kubernetes

- But why we need Container Orchestration ?
  - Production environment is cluster system
  - Microservice maintain connect was required
  - Application state-full will run on stateless architecture
  - Application scale up/Down was required
  - Many native command/shell for maintain container in production environment
  - etc

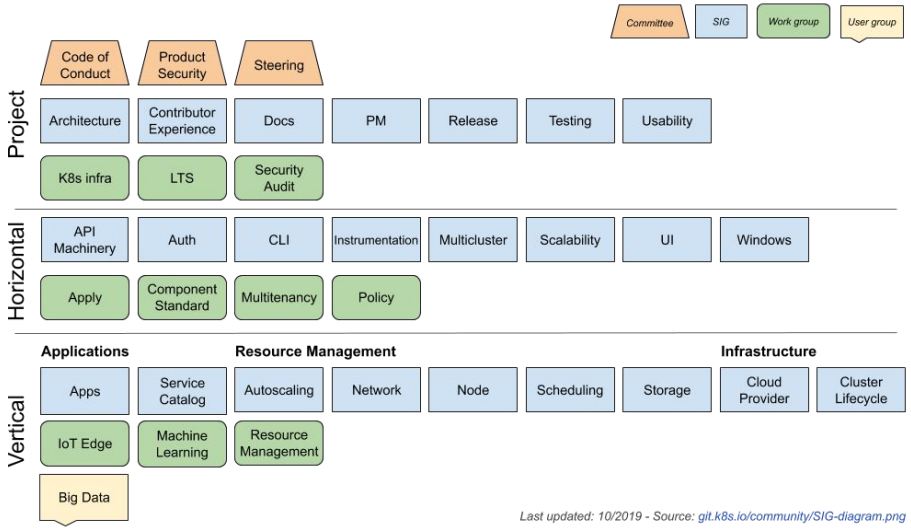
Introduction to Kubernetes

Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes

Containers, that'll fix it.

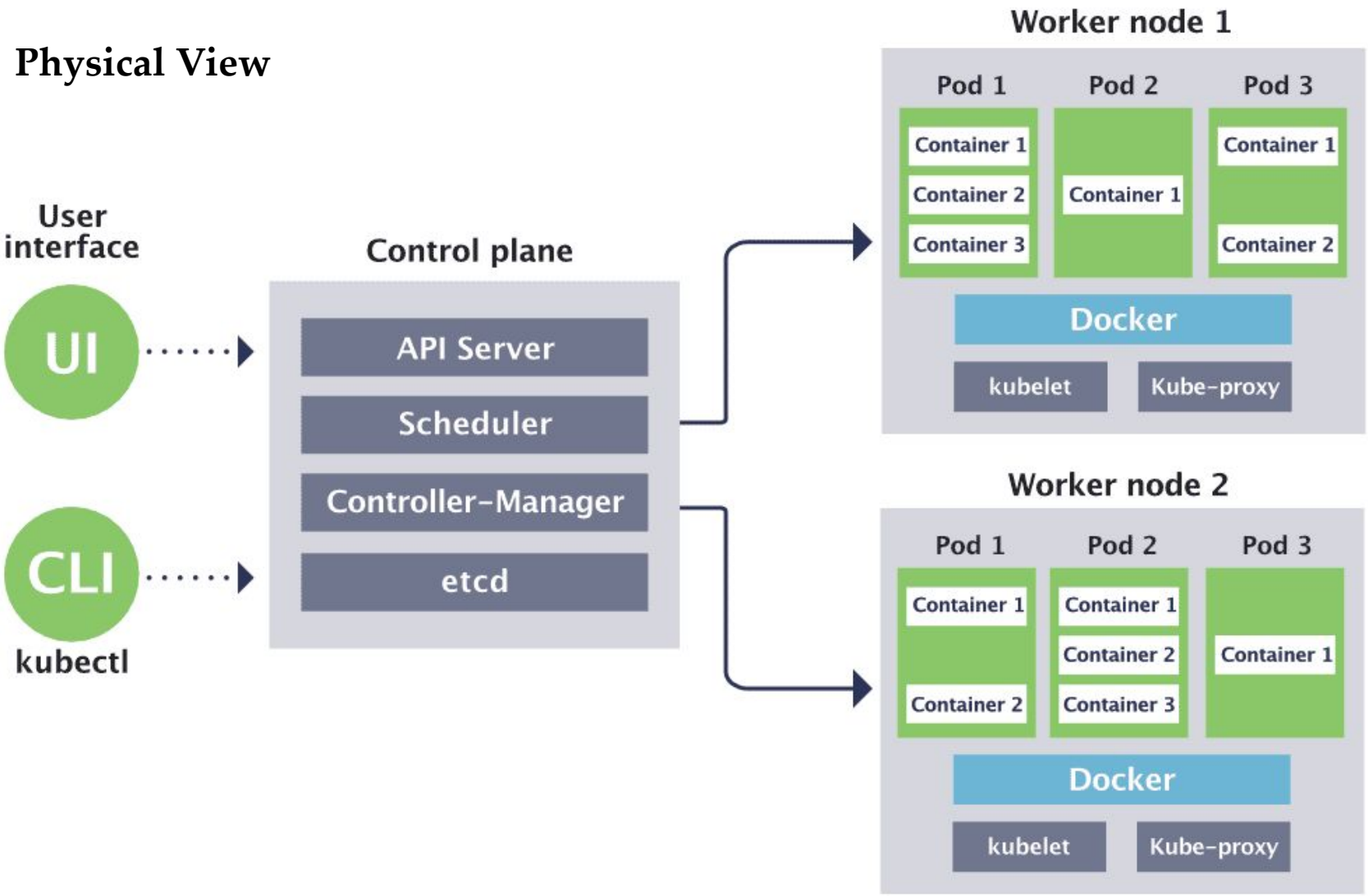


Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes

Physical View

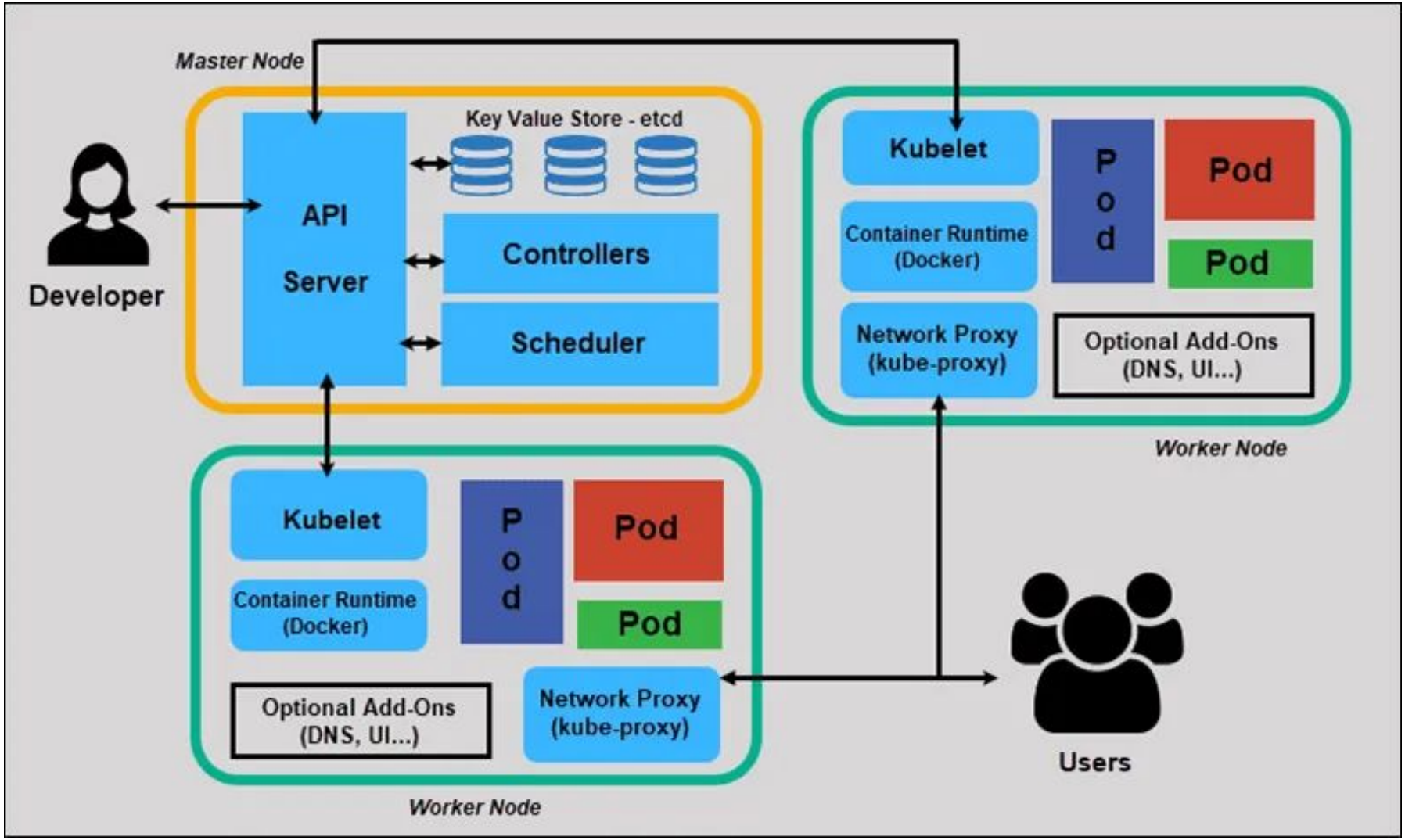


Ref: https://www.cncf.io/blog/2019/08/19/how-kubernetes-works/

Kubernetes: Production Workload Orchestration

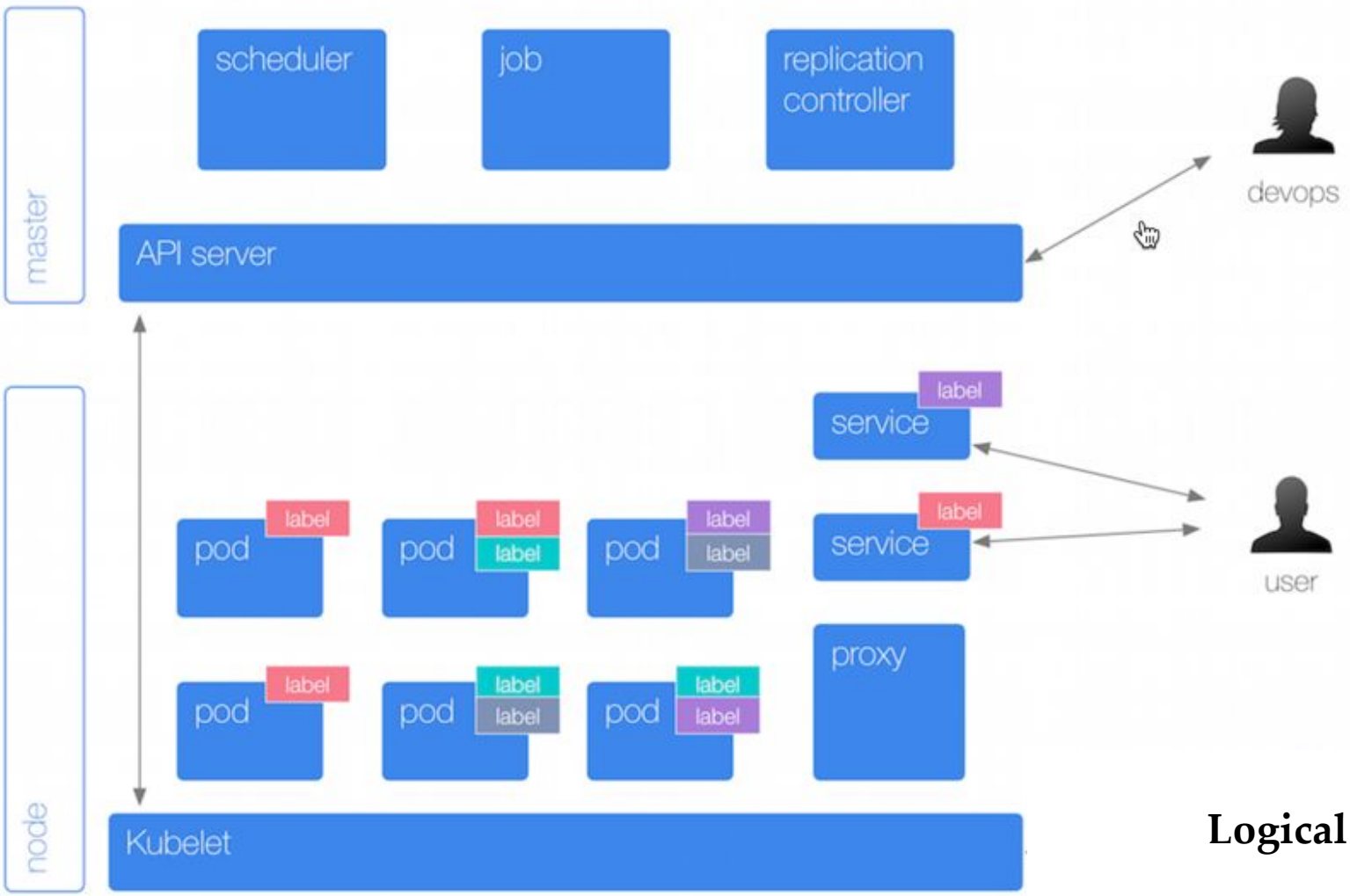
kubernetes by Google

Introduction to Kubernetes



Ref: https://phoenixnap.com/kb/understanding-kubernetes-architecture-diagrams

Introduction to Kubernetes



Logical View

Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes

- 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

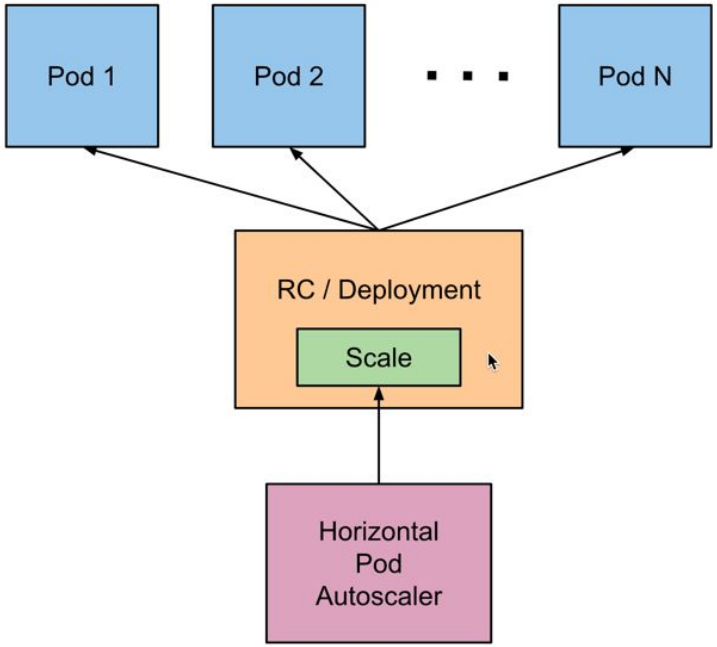
Introduction to Kubernetes

- Automatic binpacking
  - CPU/Memory's utilization can define on Pods (Smallest Unit of Kubernetes)
  - Schedule will select the node by ensure all resource is enough for running Pods as required
  - If reach memory limit
    - Current Pods will be terminate (Kill)
    - If restart flag was set. Kubenetes will try to restart Pods on other node
  - If reach cpu limit
    - Schedule will not kill Pods and waiting for it back to normal state
  - Check available node resource by command: kubectl describe node

```
Non-terminated Pods:          (3 in total)
  Namespace                   Name                          CPU Requests    CPU LimitsMemory Requests  Memory Limits
  ---------                   ----                          ------------    ----------------------------  -------------
  kube-system                 kube-addon-manager-minikube   5m (0%)         0 (0%)     50Mi (2%)       0 (0%)
  kube-system                 kube-dns-268032401-mx7s3      260m (13%)      0 (0%)     110Mi (5%)      170Mi (8%)
  kube-system                 kubernetes-dashboard-hdhrz    0 (0%)          0 (0%)     0 (0%)          0 (0%)
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  CPU Requests  CPU Limits      Memory Requests Memory Limits
  ------------  ----------      --------------- -------------
  265m (13%)    0 (0%)          160Mi (8%)      170Mi (8%)
Events:         <none>
```

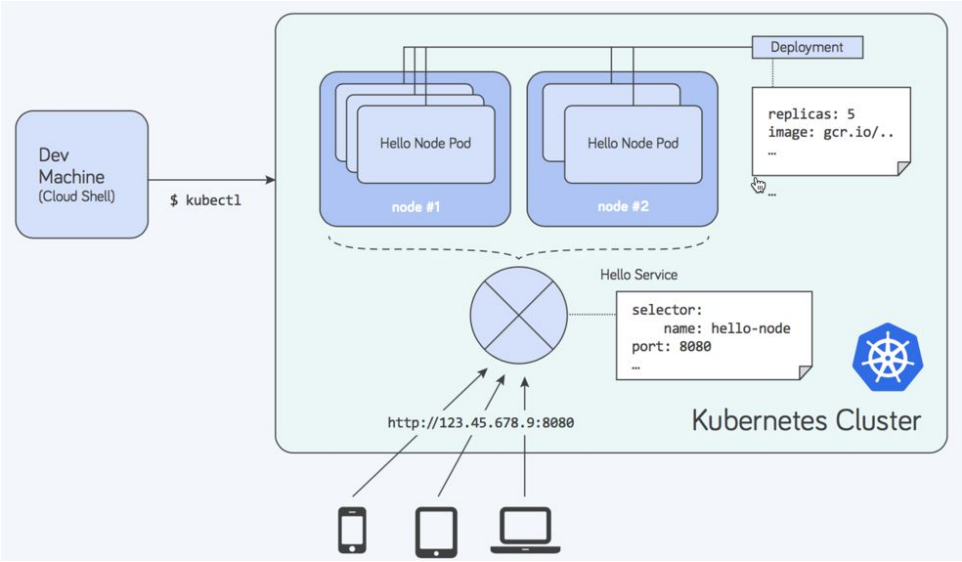
Introduction to Kubernetes

- Horizon Pods Autoscaling (hpa)
  - Talk with RC (Replication Controller) for complete task
  - Automatic scaling Pods's number base cpu's / memory utilization
  - Support multiple criteria (metric) for scale (Alpha feature)
  - Support customize criteria (metric) for scale (Alpha feature)
  - Looping check resource's utilization every 30 seconds (default)



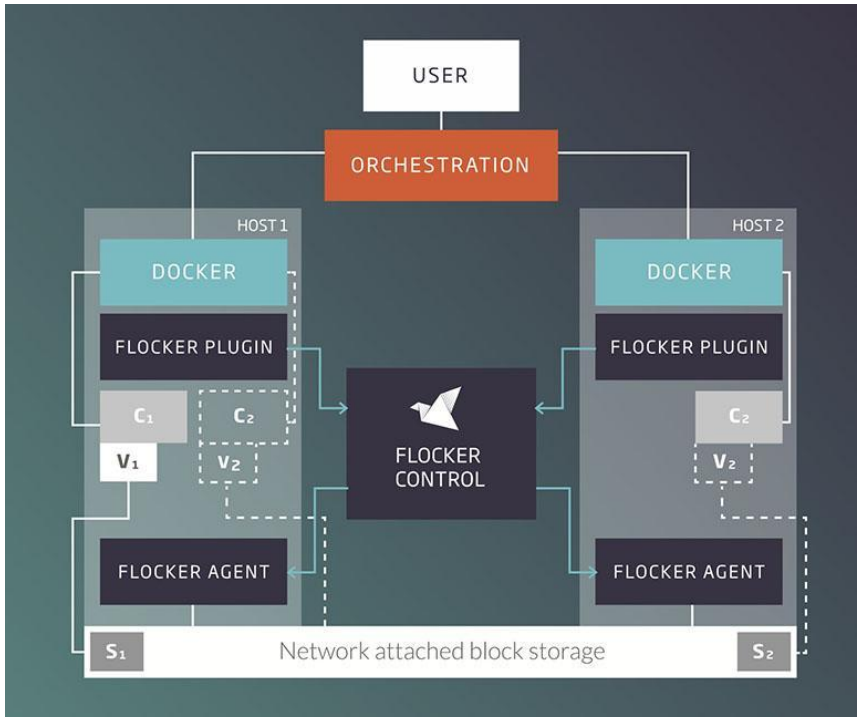
Introduction to Kubernetes

- Automated Rollout and Rollbacks
  - Update will base on Pod's template
  - Rollout existing deployment with all remain replicas number as desired
  - If rollout is success. New version will be provision until success
  - Rollback will possible with single command
  - Rollout process can pause/resume as need



Introduction to Kubernetes

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

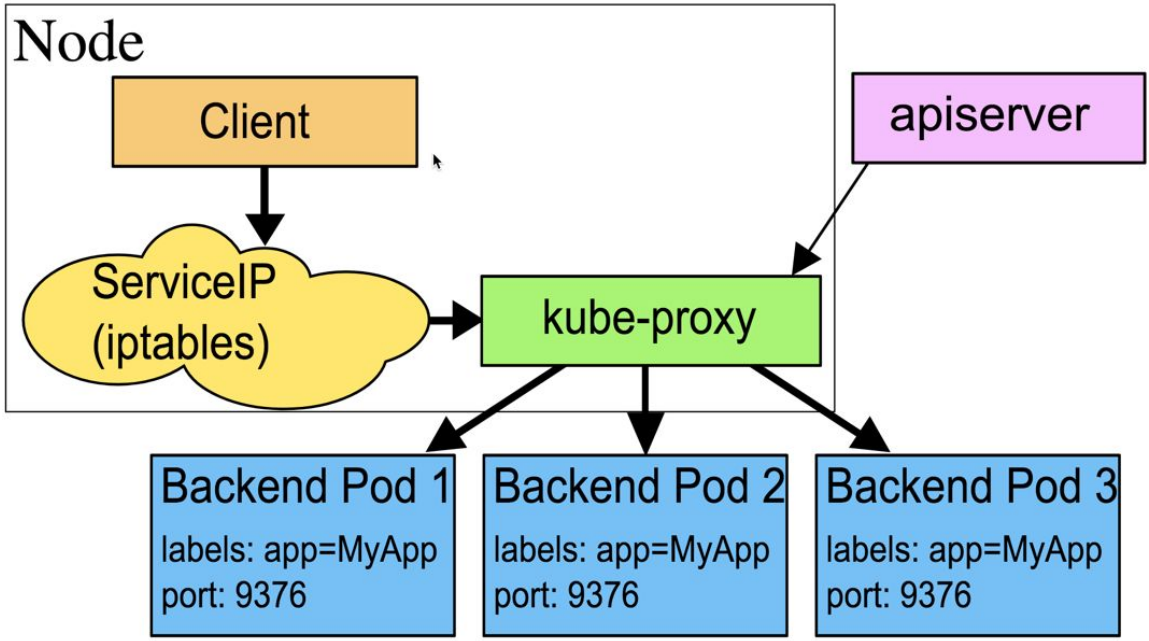


Introduction to Kubernetes

- 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

Introduction to Kubernetes

- 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)



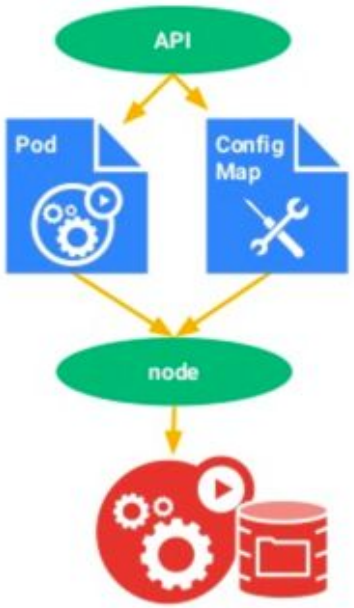
Introduction to Kubernetes

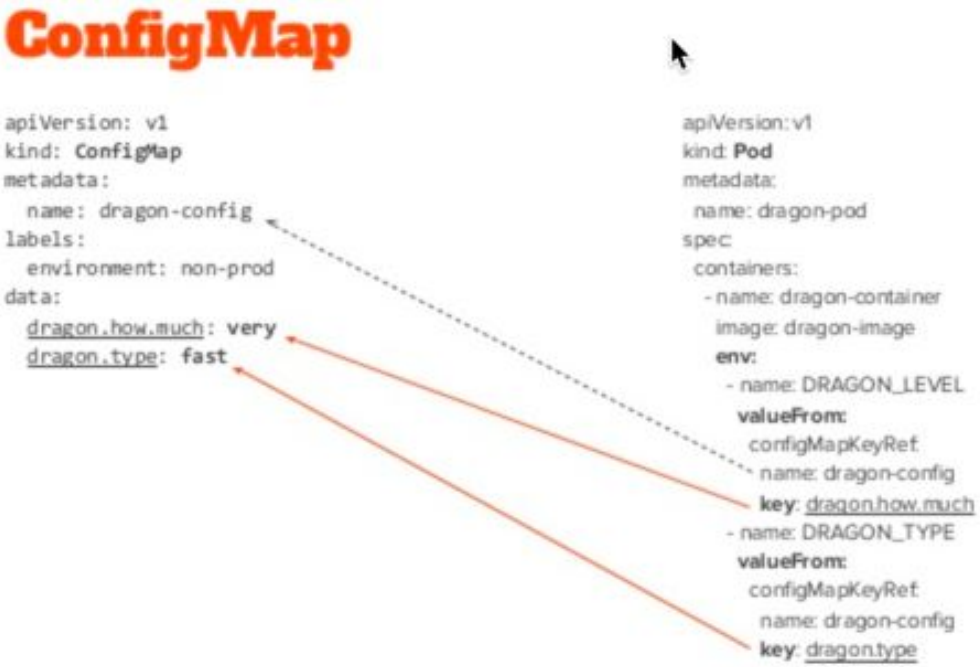
- Secret and Configuration Management
  - Kubernetes can keep confidential data (such as username/password) for running application on encrypt format.
  - Can reference on Pods instead plain text configuration.

```
$ 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
```

Introduction to Kubernetes

- Secret and Configuration Management
  - Sometimes we have a many configuration that need to specify on each application, But it should be change every time need.
  - Idea is creating configuration file (ConfigMap) for define all configuration that reference on Pods instead



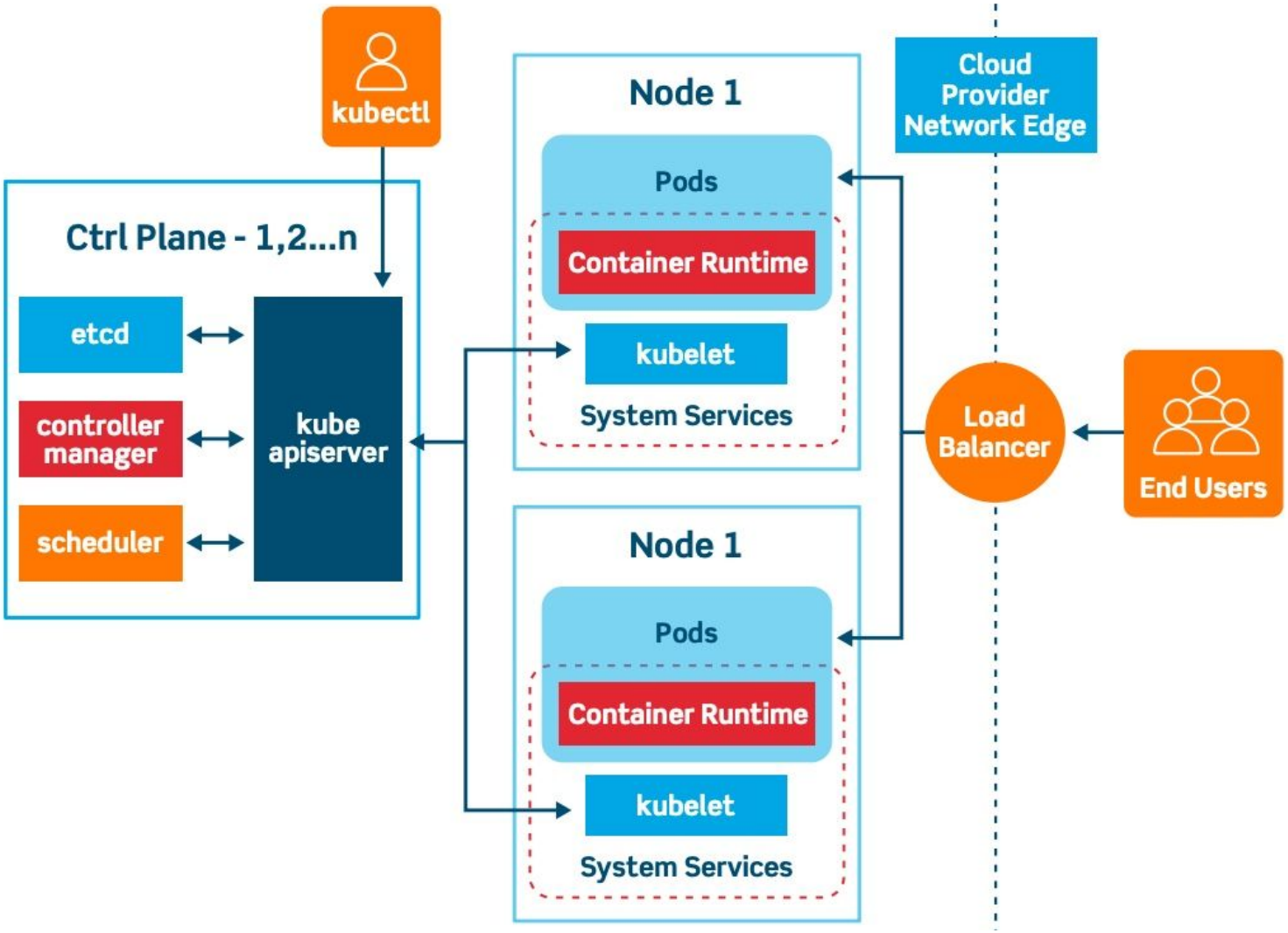


Introduction to Kubernetes

- Batch Execution
  - A job will response some kind of batch execute by create special Pods (Terminate when batch complete)
  - Normally kubernetes will use job for maintain many background process for cluster system such as Replication Controller (RC) will submit job for start new Pods when existing is fail or delete.
  - Type of Job
    - Non-parallel jobs
    - Parallel jobs with fix-completion
    - Parallel jobs with work queue

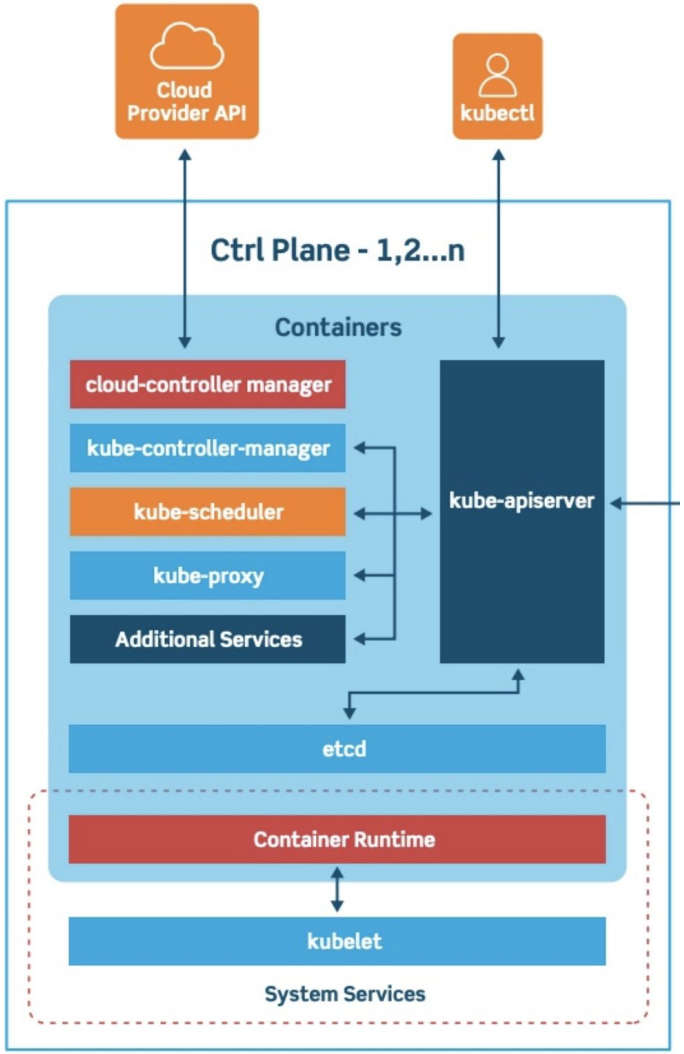
System Architecture

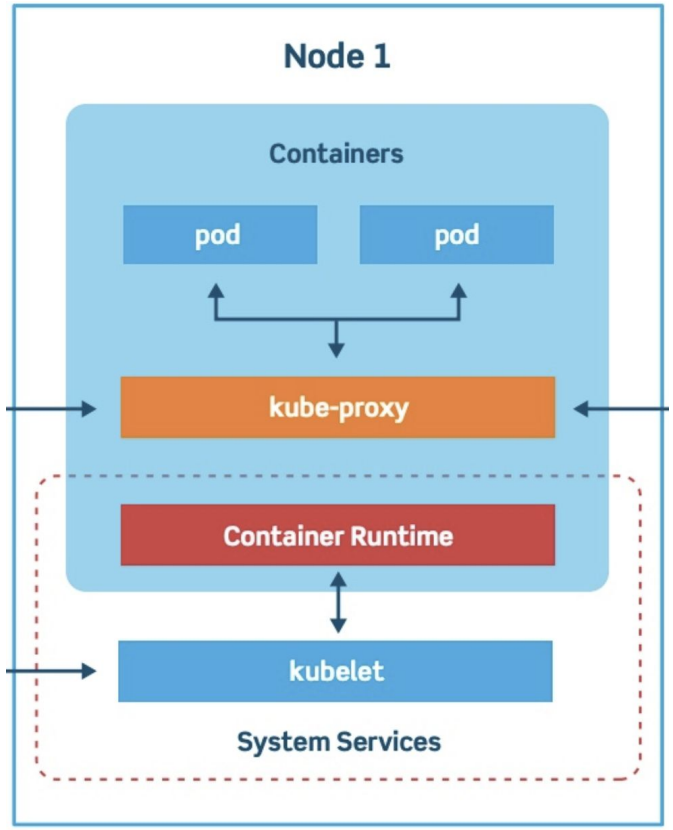
System Architecture



Ref: https://platform9.com/blog/kubernetes-enterprise-chapter-2-kubernetes-architecture-concepts/

System Architecture





Ref: https://platform9.com/blog/kubernetes-enterprise-chapter-2-kubernetes-architecture-concepts/

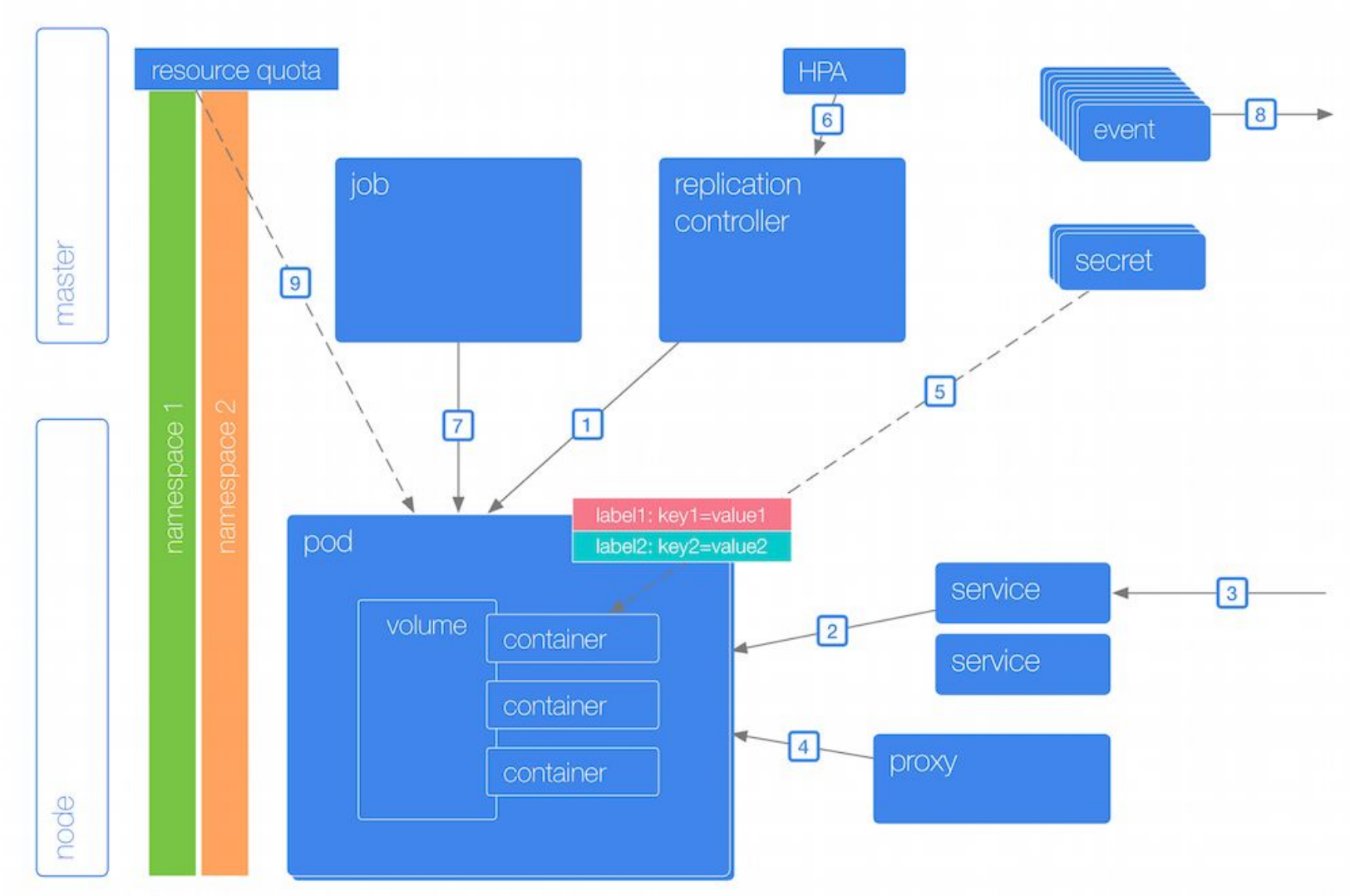
System Architecture

- Kubernetes Master
  - API Server: Interface for all UI to command that interact with kubernetes cluster
  - Scheduler (Job Dispatcher): Schedule all nodes resource and dispatching Pods to nodes with match criteria
    - CPU ?
    - Memory ?
    - Affinity / Constraint ?
  - Controller and Replication Controller (RC) : Control/Coordinate all nodes for maintain server as configure on cluster system
  - Etcd (Open-source): Key-value database for keep state of nodes/Pods/Container
  - Secret: Encrypt confidential data
  - HPA (Horizon Pods Auto scaling): Scale pods with CPU criteria (major)
  - Event: Keep log and event on cluster
  - NameSpace: Limit resource quota via define namespace (cpu, memory, pods etc)

System Architecture

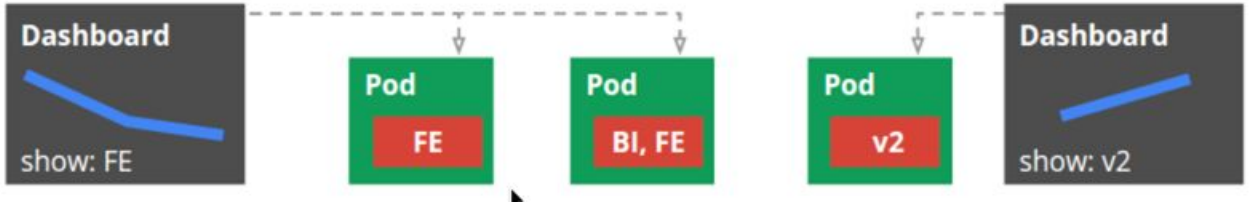
- Kubernetes Node
  - Pods: Pods is smallest deployable units of computing that can be created and manage in kubernetes (Pods != Container).
  - Container runeimt: Container engine for run container. Kubernetes support the runtime with standard CRI. such as cri-o, containerd
  - Kubelet: Agent on node to talk with kubernetes master and send status/health of node
  - Kube-Proxy: Manage all network interface on node (Core network component)

System Architecture



Ref: http://k8s.info/cs.html

System Architecture




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

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

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

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

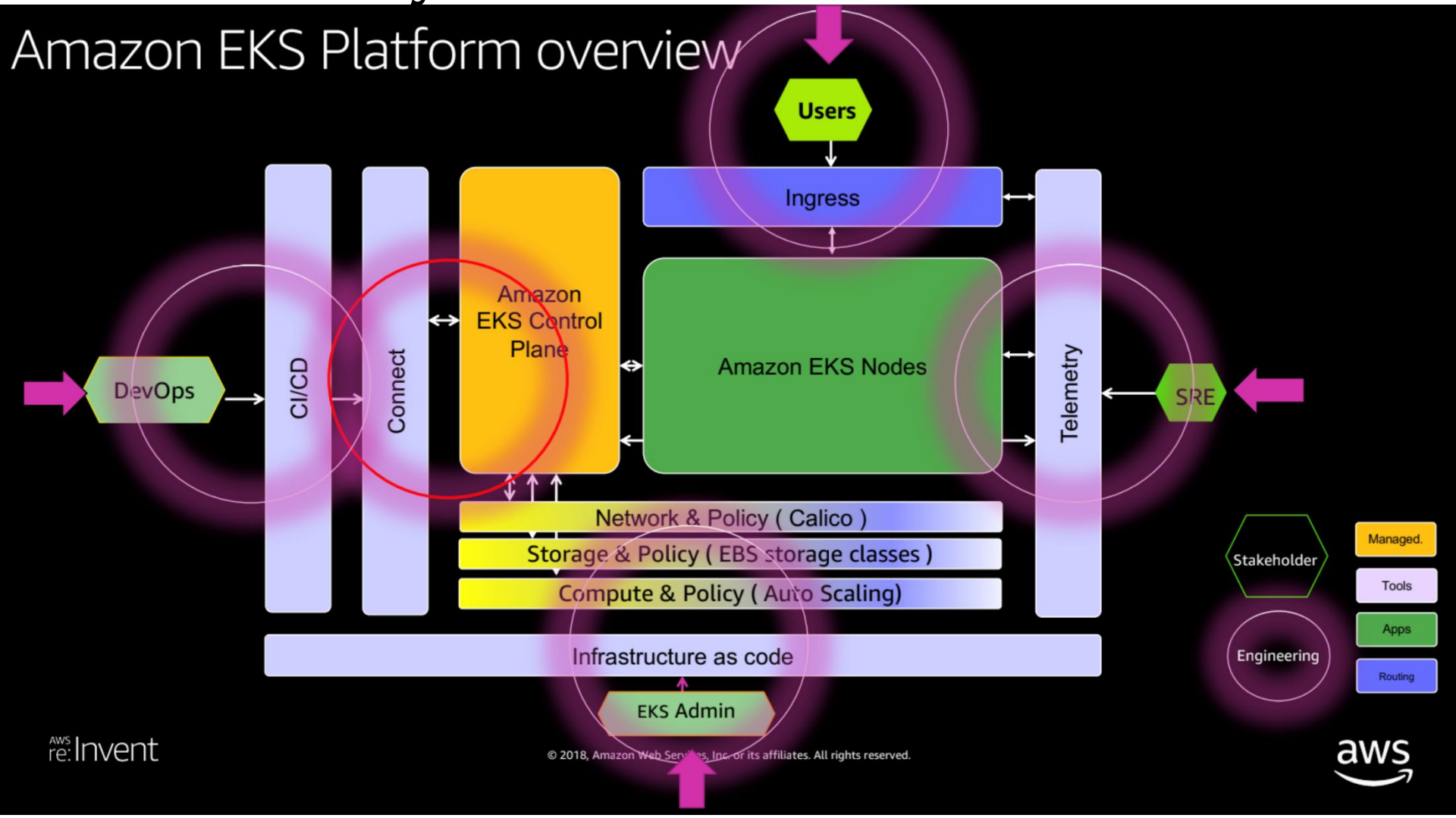
```
"partition" : "customerA", "partition" : "customerB"
```

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

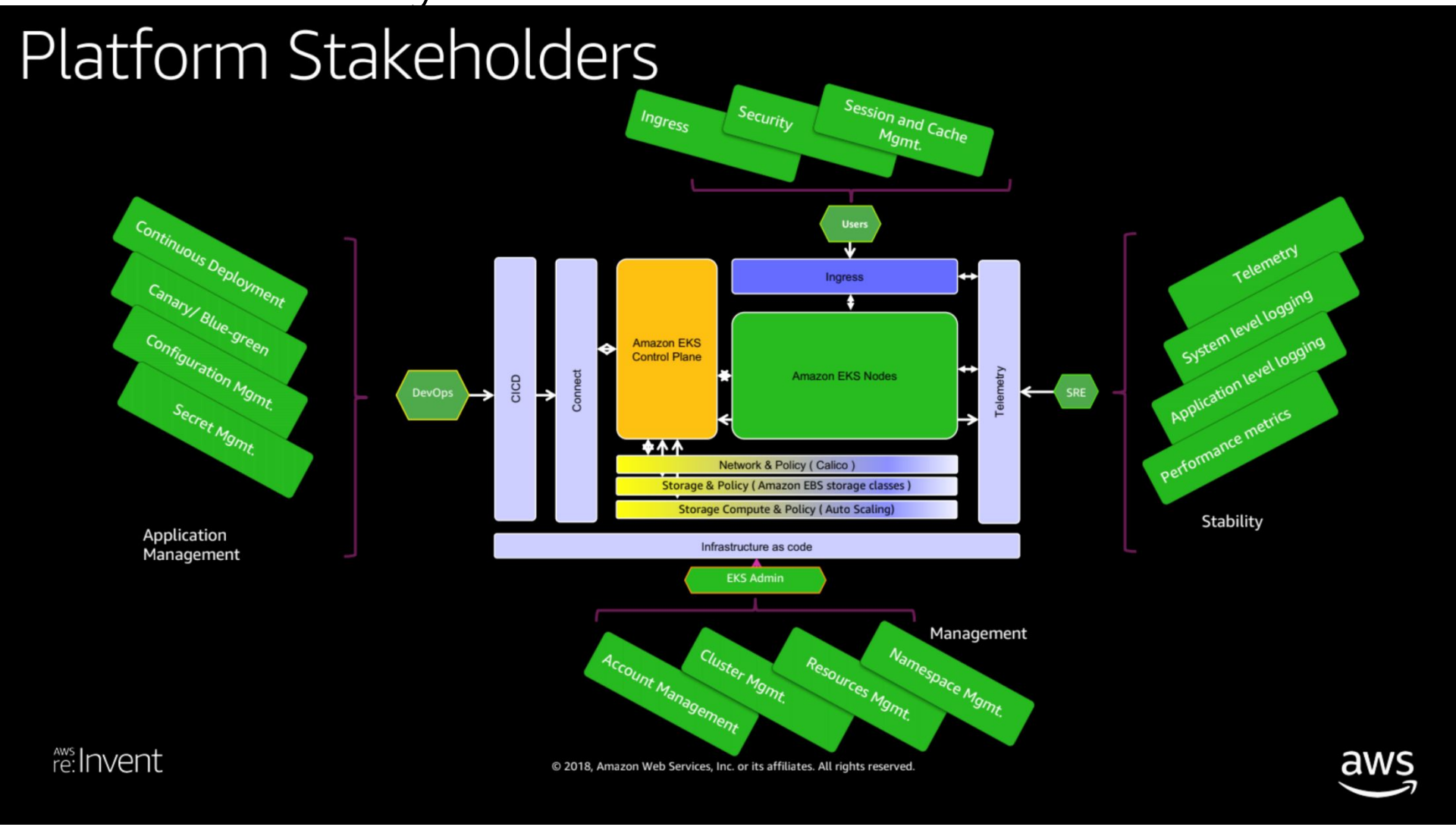
System Architecture

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

System Architecture



System Architecture



Fundamental of Kubernetes

Kubernetes: Production Workload Orchestration

kubernetes by Google

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
  - Kubernetes view point:
    - 1 Pods = 1 Container
    - 1 Pods = N Container (Container on the same context, Work closely)
    - So we can have 1 Pods for container more than 1 container

Pods, Container and Services

- Pods vs Container
  - All container on same Pods will share:
    - Process ID (PID)
    - Network access (Communicate to each other via "localhost")
    - Internal Process Command (IPC)
    - Unix Time-Sharing (UTS)
    - Hostname
    - IP Address/Ports

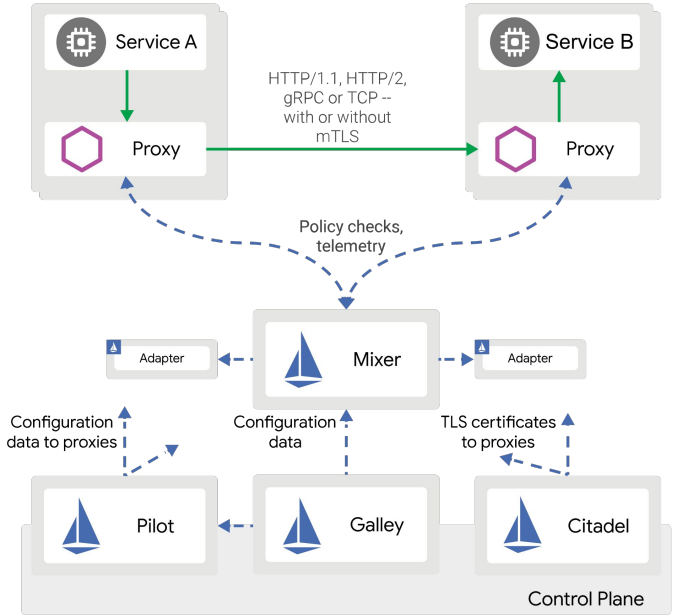
  - Use Case for Multiple Pods:
    - Apache (1 Container) +Tomcat (1 Container)
    - Apache(1 Container) + PHP (1 Container)
    - Nginx (Cache: 1 Container) + Apache/PHP (1 Container)
    - Web Server (1 Container) + Data Volume(Cache: 1 Container)
    - Service Mesh (Istio / Kong etc)

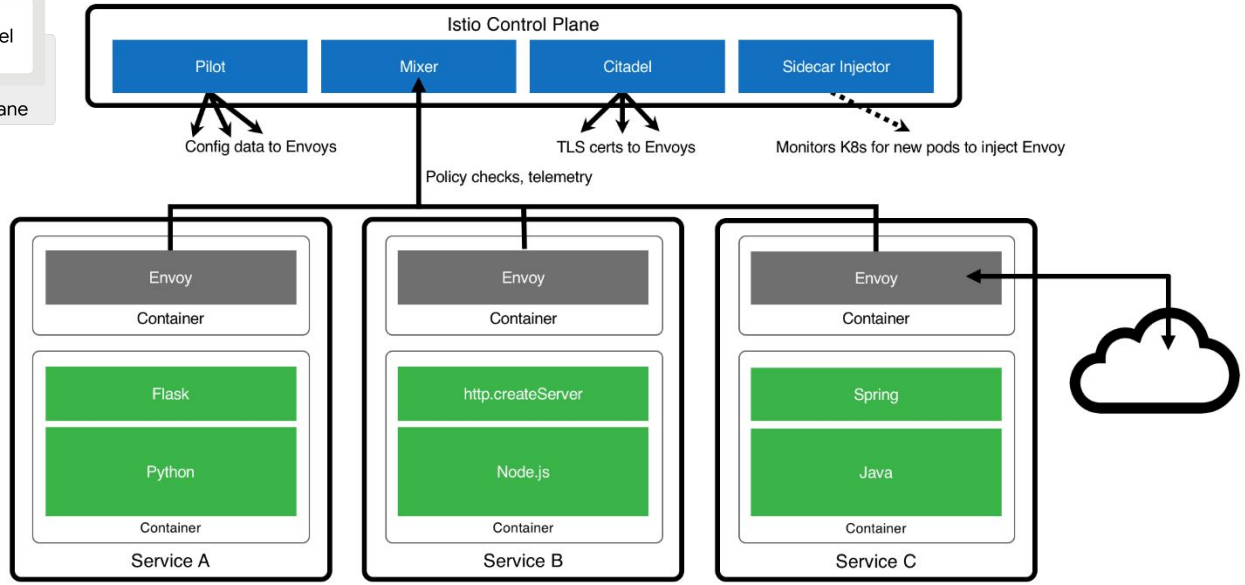
  - Pods will can create replicas of 1000+ set on cluster system

Pods, Container and Services

- Pods Life-Cycle
  - **Pending**: The Pod has been accepted by the Kubernetes cluster, but one or more of the containers has not been set up and made ready to run. This includes time a Pod spends waiting to be scheduled as well as the time spent downloading container images over the network.
  - **Running**: The Pod has been bound to a node, and all of the containers have been created. At least one container is still running, or is in the process of starting or restarting.
  - **Succeeded**: All containers in the Pod have terminated in success, and will not be restarted.
  - **Failed**: All containers in the Pod have terminated, and at least one container has terminated in failure. That is, the container either exited with non-zero status or was terminated by the system.
  - **Unknow**: For some reason the state of the Pod could not be obtained. This phase typically occurs due to an error in communicating with the node where the Pod should be running.

Pods, Container and Services



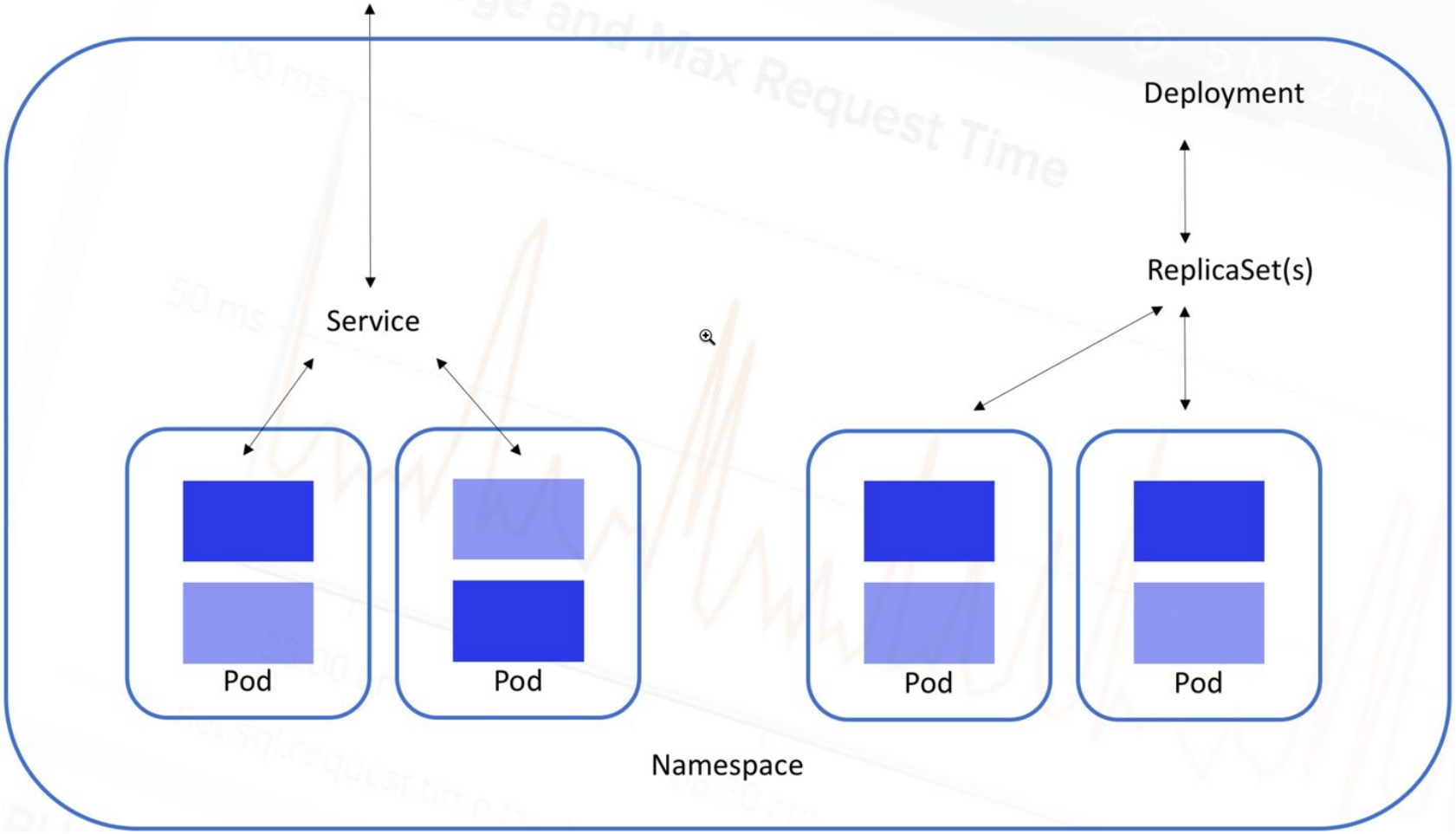


Kubernetes: Production Workload Orchestration

kubernetes by Google

Pods, Container and Services

- Container/ Pods / Service / Deployment/ RS



Pods, Container and Services

- 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>

```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
```

Pods, Container and Services

- Imperative Command:
  - Create object via interactive shell
    - "run": create new pods
    - "expose": create new service to access pods
    - "autoscale": create autoscale to deployment etc.
    - "create": for create specific object
      - kubectl create <objecttype> [<subtype>] <instancename>
      - Ex: "kubectl create service nodeport myservice"
  - Update object via interactive shell
    - "scale" add/remove replicas of pods
    - "annotate" add/remove annotation
    - "label" add/remove label
    - "set" <field> define some specific value
      - Ex: kubectl set env deployment/registry ac=/local
    - "edit" for direct edit yaml file on k8s system
    - "patch" for update some field directly on single command

Pods, Container and Services

- Imperative Command:
  - Delete object via interactive shell
    - “delete”: delete element by command
      - Ex: kubectl delete pods/abandon
      - Ex: kubectl delete service/abdc

Pods, Container and Services

- 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 !!!**

Pods, Container and Services

- 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
```

Pods, Container and Services

- Imperative commands:

Pods, Container and Services

- Imperative object configuration

kubectl create –f <Filename>

- Create Pods (YAML)

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

Create Service (YAML)

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

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

Pods, Container and Services

Code Beautify

JSON Formatter | My Ip | Search | Recent Links | Sample | More | Sign in | (?)

YAML Converter

Save & Share

YAML Input sample

```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
  nodeSelector:
    kubernetes.io/hostname: kubernetes-1
```

Load Url

Browse

YAML TO JSON

YAML TO XML

YAML TO CSV

Validate

Seen this ad multiple times

Not interested in this ad

Download

Result : YAML TO JSON

```json
{
    "apiVersion": "v1",
    "kind": "Pod",
    "metadata": {
        "name": "webtest",
        "labels": {
            "name": "web",
            "owner": "Praparn_L",
            "version": "1.0",
            "module": "WebServer",
            "environment": "development"
        }
    },
    "spec": {
        "containers": [
            {
                "name": "webtest",
                "image": "labdocker/cluster
                    :webservicelite",
                "ports": [
                    {
                        "containerPort": 5000,
                        "protocol": "TCP"
                    }
                ]
            }
        ],
        "nodeSelector": {
            "kubernetes.io/hostname": "kubernetes-1"
        }
    }
}
```

Pods, Container and Services

kube-score.com

Mac Ked - Mac... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | Taiwan | NGINX | Kubernetes | PWA_Progressive_... | MYSQL_Cluster | GeneralKB

kube-score

Kubernetes object analysis with recommendations for improved reliability and security.

`kube-score` is a tool that does static code analysis of your Kubernetes object definitions. The output is a list of recommendations of what you can improve to make your application more secure and resilient.

`kube-score` is open-source and available under the MIT-license. For more information about how to use kube-score, see zegl/kube-score on GitHub. Use this website to easily test kube-score, just paste your object definition YAML or JSON in the box below.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: statefulset-test-1
spec:
  template:
    metadata:
      labels:
        app: foo
    spec:
      containers:
      - name: foobar
        image: foo:bar
---
apiVersion: policy/v1beta1
kind: PodDisruptionBudget
```

```
apps/v1/Deployment statefulset-test-1
    [CRITICAL] Container Image Pull Policy
        · foobar -> ImagePullPolicy is not set to Always
            It's recommended to always set the ImagePullPolicy to Always, to make sure that the
            imagePullSecrets are always correct, and to always get the image you want.
    [CRITICAL] Pod NetworkPolicy
        · The pod does not have a matching NetworkPolicy
            Create a NetworkPolicy that targets this pod to control who/what can communicate with this pod.
            Note, this feature needs to be supported by the CNI implementation used in the Kubernetes cluster
            to have an effect.
    [CRITICAL] Container Security Context
        · foobar -> Container has no configured security context
            Set securityContext to run the container in a more secure context.
```

Ref: https://github.com/zegl/kube-score#installation

Pods, Container and Services

README.md

Conftest

go report A+ | netlify Success

Conftest helps you write tests against structured configuration data. Using Conftest you can write tests for your Kubernetes configuration, Tekton pipeline definitions, Terraform code, Serverless configs or any other config files.

Conftest uses the Rego language from Open Policy Agent for writing the assertions. You can read more about Rego in How do I write policies in the Open Policy Agent documentation.

Here's a quick example. Save the following as `policy/deployment.rego`:

```
package main

deny[msg] {
  input.kind == "Deployment"
  not input.spec.template.spec.securityContext.runAsNonRoot

  msg := "Containers must not run as root"
}

deny[msg] {
  input.kind == "Deployment"
  not input.spec.selector.matchLabels.app

  msg := "Containers must provide app label for pod selectors"
}
```

Assuming you have a Kubernetes deployment in `deployment.yaml` you can run Conftest like so:

```
$ conftest test deployment.yaml
FAIL - deployment.yaml - Containers must not run as root
FAIL - deployment.yaml - Containers must provide app label for pod selectors

2 tests, 0 passed, 0 warnings, 2 failures, 0 exceptions
```

Conftest isn't specific to Kubernetes. It will happily let you write tests for any configuration files in a variety of different formats. See the documentation for installation instructions and more details about the features.

Want to contribute to Conftest?

- See DEVELOPMENT.md to build and test Conftest itself.
- See CONTRIBUTING.md to get started.

For discussions and questions join us on the Open Policy Agent Slack in the `#conftest` channel.

check_image_registry.rego

```
package main

deny[msg] {

  input.kind == "Deployment"
  image := input.spec.template.spec.containers[_].image
  not startswith(image, "my-company.com/")
  msg := sprintf("image '%v' doesn't come from my-company.com repository",
}
```

bash

```
$ conftest test --policy ./conftest-checks base-valid.yaml
FAIL - base-valid.yaml - image 'hashicorp/http-echo' doesn't come from my-c
1 tests, 1 passed, 0 warnings, 1 failure

$
```

Ref: https://github.com/open-policy-agent/conftest

Pods, Container and Services

README.md

KubeLibrary

circleci passing | pypi v0.4.0 | downloads 2.7k/month | license MIT | slack robotframework/#kubernetes

RobotFramework library for testing Kubernetes cluster

Quick start

```
# install library itself
pip install robotframework-kubelibrary

# export KUBECONFIG
export KUBECONFIG=~/.kube/config

# run example tests
pip install robotframework-requests
git clone https://github.com/devopsspiral/KubeLibrary.git
cd KubeLibrary
robot -e prerelease testcases
```

Example testcase

```
Pods in kube-system are ok
    [Documentation]  Test if all pods in kube-system initiated correctly and are running or succeeded
    [Tags]    cluster    smoke
    Given kubernetes API responds
    When getting all pods in  "kube-system"
    Then all pods in "kube-system" are running or succeeded

Grafana has correct version
    [Documentation]  Test if Grafana container image is in correct version
    [Tags]    grafana
    Given kubernetes API responds
    When accessing "grafana-" excluding "svclb" container images version in "default"
    Then "grafana/grafana:6.5.0" version is used
```

More examples in testcases/ directory

KubeLibrary

RobotFramework library for testing Kubernetes cluster

Ref: https://github.com/zegl/kube-score#installation

Pods, Container and Services

- Imperative object configuration

kubectl create –f <Filename>

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

Pods, Container and Services

- Imperative object configuration

kubectl create –f <Filename>

Pods, Container and Services

- Check log on container:

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

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

- Shell inside container:

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

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

Pods, Container and Services

- Check detail property of Pods/Service

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

```
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl describe pods webtest
Name:           webtest
Namespace:      default
Node:           minikube/192.168.99.101
Start Time:     Sun, 02 Jul 2017 12:56:50 +0700
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
Annotations:    <none>
Status:         Running
IP:             172.17.0.4
Controllers:    <none>
Containers:
  webtest:
    Container ID:       docker://04b9cdfdd6451a4e78c873f704fb4d35de7bba082343d0773c7ea0ebbbd3f03a
    Image:              labdocker/cluster:webservicelite
    Image ID:           docker://sha256:837c8f41c918ede06705f9b554b6120fdb654cc2a8eb7eec74bc383b09865f2b
    Port:               5000/TCP
    State:              Running
      Started:          Sun, 02 Jul 2017 12:56:51 +0700
    Ready:              True
    Restart Count:      0
    Environment:        <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-vxm58 (ro)
Conditions:
  Type          Status
  Initialized   True
  Ready         True
  PodScheduled  True
Volumes:
  default-token-vxm58:
    Type:       Secret (a volume populated by a Secret)
    SecretName: default-token-vxm58
    Optional:   false
QoS Class:      BestEffort
Node-Selectors: <none>
Tolerations:    <none>
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type            Reason          Message
  ---------     --------        -----   ----                    -------------                   --------        ------          -------
  14m           14m             1       default-scheduler                                       Normal          Scheduled       Successfully assigned webtest to minikube
  14m           14m             1       kubelet, minikube       spec.containers{webtest}        Normal          Pulled          Container image "labdocker/cluster:webservicelite" alread
y present on machine
  14m           14m             1       kubelet, minikube       spec.containers{webtest}        Normal          Created         Created container with id 04b9cdfdd6451a4e78c873f704fb4d3
5de7bba082343d0773c7ea0ebbbd3f03a
  14m           14m             1       kubelet, minikube       spec.containers{webtest}        Normal          Started         Started container with id 04b9cdfdd6451a4e78c873f704fb4d3
5de7bba082343d0773c7ea0ebbbd3f03a
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$
```

Pods, Container and Services

- Check detail property of Pods/Service

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

```
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl describe svc webtest
Name:                     webtest
Namespace:                default
Labels:                   environment=development
                          module=WebServer
                          name=web
                          owner=Praparn_L
                          version=1.0
Annotations:              <none>
Selector:                 environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
Type:                     NodePort
IP:                       10.0.0.87
Port:                     http    5000/TCP
NodePort:                 http    32131/TCP
Endpoints:                172.17.0.4:5000
Session Affinity:         None
Events:                   <none>
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
```

Pods, Container and Services

Kubectl reference guide:

kubernetes

Documentation Kubernetes Blog

Search this site

- kubectl reference
  - kubectl
  - kubectl annotate
  - kubectl api-resources
  - kubectl api-versions
  - kubectl apply
  - kubectl attach
  - kubectl auth
  - kubectl autoscale
  - kubectl certificate
  - kubectl cluster-info
  - kubectl completion
  - kubectl config
  - kubectl cordon
  - kubectl cp
  - kubectl create
  - kubectl debug
  - kubectl delete
  - kubectl describe
  - kubectl diff
  - kubectl drain
  - kubectl edit
  - kubectl events
  - kubectl exec
  - kubectl explain
  - kubectl expose
  - kubectl get
  - kubectl kustomize
  - kubectl label
  - kubectl logs
  - kubectl options
  - kubectl patch
  - kubectl plugin

Kubernetes Documentation / Reference / Command line tool (kubectl) / kubectl reference

kubectl reference

kubectl

kubectl annotate

kubectl api-resources

kubectl api-versions

kubectl apply

kubectl attach

kubectl auth

kubectl autoscale

kubectl certificate

kubectl cluster-info

kubectl completion

kubectl config

kubectl cordon

kubectl cp

kubectl create

kubectl apply set-last-applied

Synopsis

Set the latest last-applied-configuration annotations by setting it to match the contents of a file. This results in the last-applied-configuration being updated as though 'kubectl apply -f<file> ' was run, without updating any other parts of the object.

```
kubectl apply set-last-applied -f FILENAME
```

Examples

```
# Set the last-applied-configuration of a resource to match the contents of a file
kubectl apply set-last-applied -f deploy.yaml

# Execute set-last-applied against each configuration file in a directory
kubectl apply set-last-applied -f path/

# Set the last-applied-configuration of a resource to match the contents of a file; will create
kubectl apply set-last-applied -f deploy.yaml --create-annotation=true
```

kubectl apply view-last-applied

Synopsis

View the latest last-applied-configuration annotations by type/name or file.

The default output will be printed to stdout in YAML format. You can use the -o option to change the output format.

```
kubectl apply view-last-applied (TYPE [NAME | -l label] | TYPE/NAME | -f FILENAME)
```

Examples

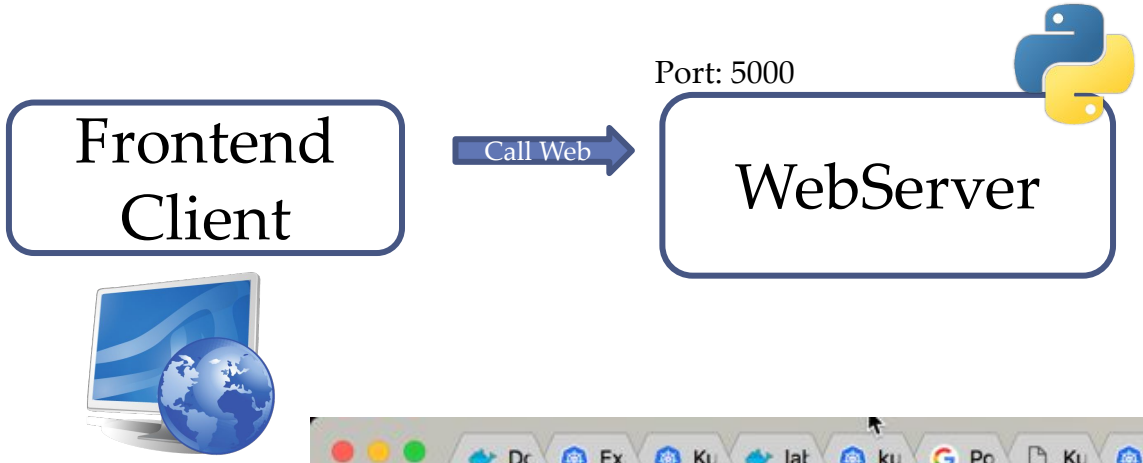
```
# View the last-applied-configuration annotations by type/name in YAML
kubectl apply view-last-applied deployment/nginx

# View the last-applied-configuration annotations by file in JSON
kubectl apply view-last-applied -f deploy.yaml -o json
```

Ref:https://kubernetes.io/docs/reference/kubectl/generated/

kubernetes by Google

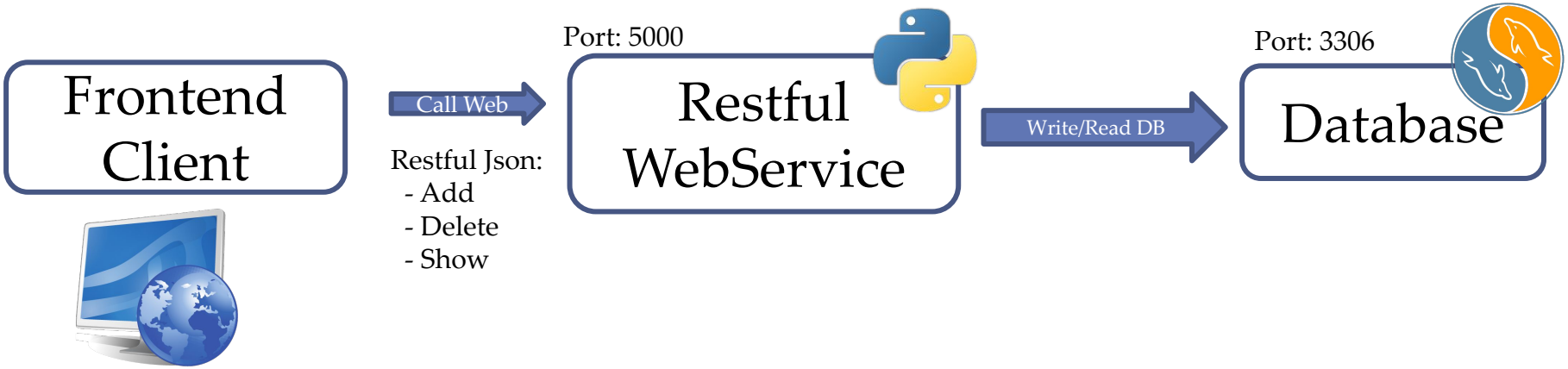
Workshop: Pods & Service

- Part 1: Deploy simple web pods with single container



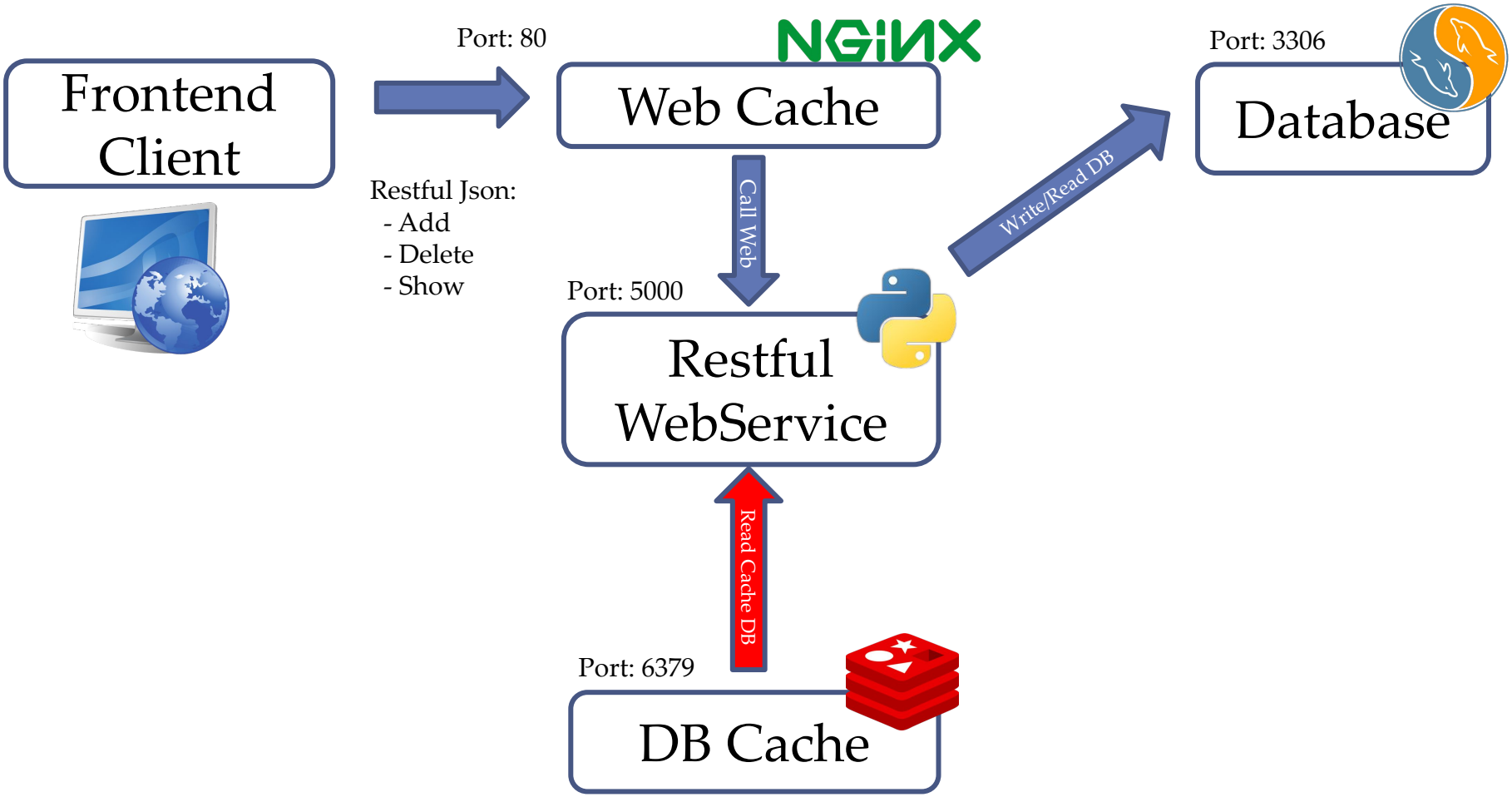
Pods, Container and Service

- Example Case: Basic Restful API



Pods, Container and Service

- Example Case: High I/O Restful API



kubernetes by Google

Pods, Container and Service

- Restful WebService
  - /init

```
@app.route('/init')
def init():
    MAIN_DB.execute("DROP DATABASE IF EXISTS ACCTABLE")
    MAIN_DB.execute("CREATE DATABASE ACCTABLE")
    MAIN_DB.execute("USE ACCTABLE")
    sql = """CREATE TABLE users (
        ID int,
        USER char(30),
        DESCRIPE char(250)
     )"""
    MAIN_DB.execute(sql)
    db.commit()
    return "###########   Database Create New Account Table Done  ###########"
```

  - /insertuser

```
@app.route("/users/insertuser", methods=['POST'])
def add_users():
    req_json = request.get_json()
    MAIN_DB.execute("INSERT INTO ACCTABLE.users (ID, USER, DESCRIPE) VALUES (%s,%s,%s)", (req_json['uid'], req_json['user'], req_json['descripe']))
    #curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"System Engineer"}' http://<IP
    db.commit()
    return Response("########### Record was added ###########", status=200, mimetype='application/json')
```

Pods, Container and Service

- Restful WebService
  - /removeuser/<uid>

```
@app.route('/users/removeuser/<uid>')
def remove_users(uid):
    hash = hashlib.sha224(str(uid)).hexdigest()
    key = "sql_cache:" + hash
    MAIN_DB.execute("DELETE FROM ACCTABLE.users WHERE ID =" + str(uid))
    db.commit()
    #curl http://<IP Host>:<Port>/users/removeuser/<uid>
    if (CACHE_DB.get(key)):
        CACHE_DB.delete(key)
        return Response("########### Record was deleted (Both Database Cache) ###########", status=200, mimetype='application/json')
    else:
        return Response("########### Record was deleted ###########", status=200, mimetype='application/json')
```

Pods, Container and Service

- Restful WebService
  - /users/<uid>

```
@app.route('/users/<uid>')
def get_users(uid):
    hash = hashlib.sha224(str(uid)).hexdigest()
    key = "sql_cache:" + hash
    #curl http://<IP Host>:<Port>/users/<uid>
    if (CACHE_DB.get(key)):
        return CACHE_DB.get(key) + "(Database Cache)"
    else:
        MAIN_DB.execute("select USER from ACCTABLE.users where ID=" + str(uid))
        data = MAIN_DB.fetchone()
        if data:
            CACHE_DB.set(key,data[0])
            CACHE_DB.expire(key, 36);
            return CACHE_DB.get(key)
        else:
            return "########### Record not found ###########"
```

Pods, Container and Service

- Web Cache

```
http {
    client_max_body_size 500M;
    client_body_timeout 3000s;
    include /etc/nginx/mime.types;
    default_type application/octet-stream;
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
    '$status $body_bytes_sent "$http_referer" '
    '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;
    tcp_nopush on;

    keepalive_timeout 65;

    gzip on;

    include /etc/nginx/conf.d/*.conf;
server {
        listen 80;
        client_body_buffer_size 50M;
        index index.html        index.htm;
        location / {
        proxy_pass http://webservice:5000;
        proxy_next_upstream error timeout invalid_header http_500 http_502 http_503 http_504;
        proxy_redirect off;
        proxy_buffering off;
        proxy_set_header        Host            $host;
        proxy_set_header        X-Real-IP       $remote_addr;
        proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
        }
        }
```

Pods, Container and Service

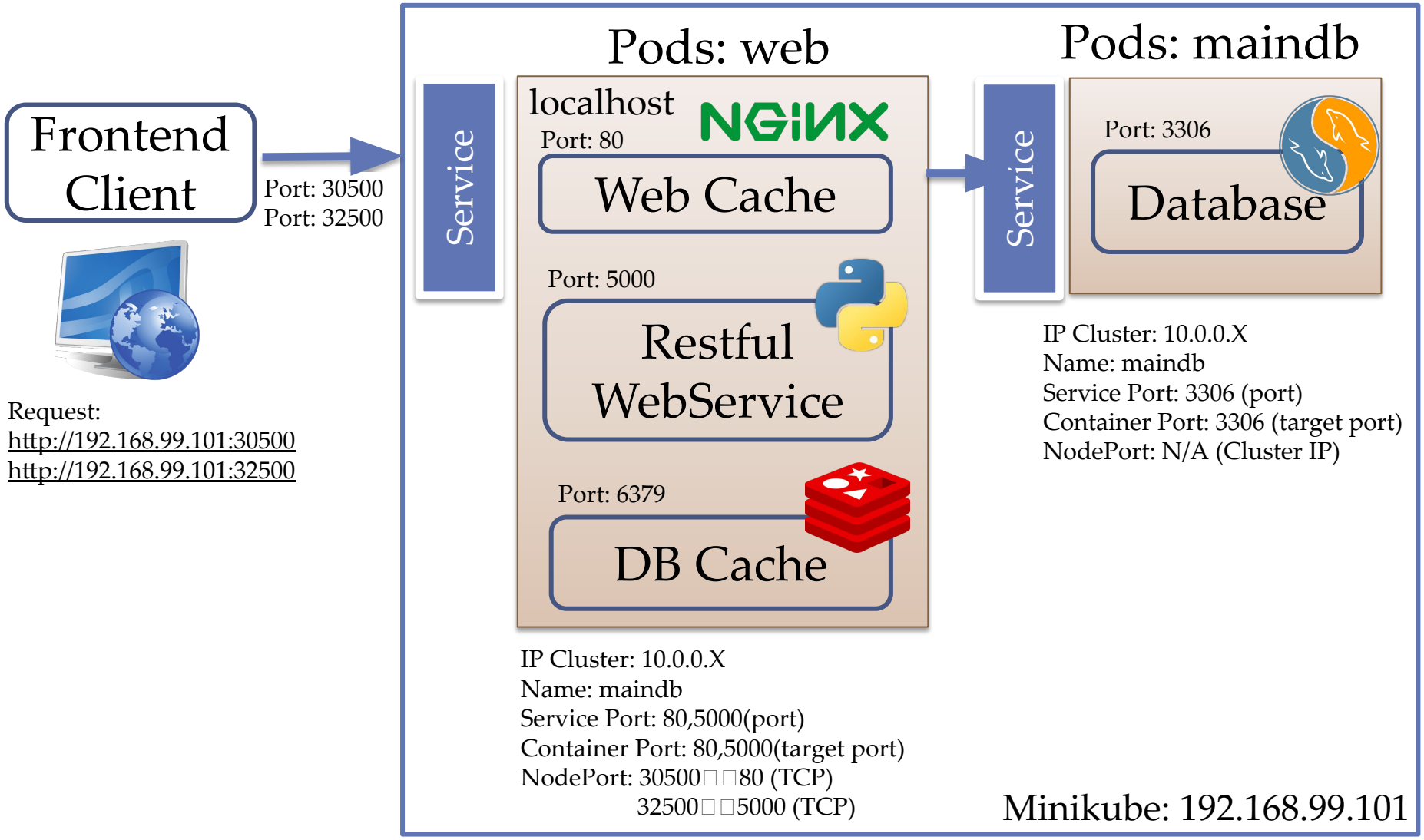
- Database and Database Cache

```yaml
maindb:
  image: labdocker/mysql:latest
  container_name: maindb
  environment:
      MYSQL_ROOT_PASSWORD: password

cachedb:
  image: labdocker/redis:latest
  container_name: cachedb

webservice:
  build: .
  dockerfile: dockerfile_python
  container_name: webservice
  ports:
   - "5000:5000"
  links:
   - cachedb:cachedb
   - maindb:maindb

webcache:
  build: .
  dockerfile: dockerfile_nginx
  container_name: webcache
  ports:
   - "80:80"
  links:
   - webservice:webservice
```

Mysql Database

Redis Key-Value Database

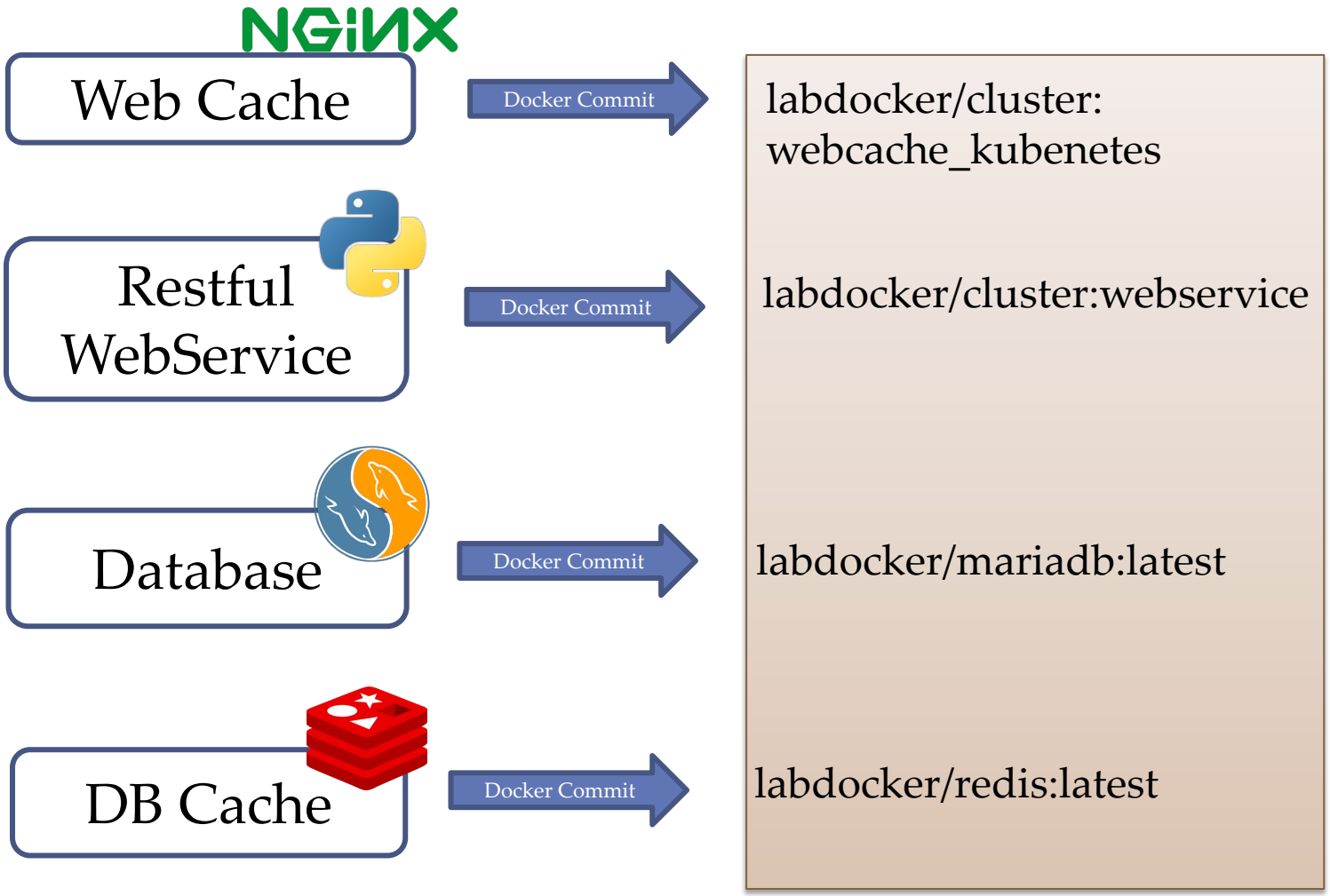
Pods, Container and Service



Kubernetes: Production Workload Orchestration

kubernetes by Google

Pods, Container and Service



Pods, Container and Service

- Pods: maindb(YML)

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

Service: maindb(YML)

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

Pods, Container and Service

- Pods: web(YML)

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: web
  labels:
    name: "web"
    owner: "Praparn_L"
    version: "1.0"
    module: "web"
    environment: "development"
spec:
  containers:
    - name: cachedb
      image: labdocker/redis:latest
      ports:
        - containerPort: 6379
          protocol: TCP
    - name: webservice
      image: labdocker/cluster:webservicev2
      env:
        - name: "REDIS_HOST"
          value: "localhost"
      ports:
        - containerPort: 5000
          protocol: TCP
    - name: webcache
      image: labdocker/cluster:webcache_kubernetes
      ports:
        - containerPort: 80
          protocol: TCP
```

Service: web (YML)

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

Pods, Container and Service

- Create "maindb" Pods and service

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
No resources found.
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
praparns-MacBook-Pro:multicontainer praparn$ kubectl create -f databasemodule_pod.yml
pod "maindb" created
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME      READY     STATUS    RESTARTS   AGE
maindb    1/1       Running   0          9s
praparns-MacBook-Pro:multicontainer praparn$ kubectl create -f databasemodule_svc.yml
service "maindb" created
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   7s
praparns-MacBook-Pro:multicontainer praparn$
```

Pods, Container and Service

- Create “web” Pods and service

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl create -f webmodule_pod.yml
pod "web" created
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME      READY     STATUS    RESTARTS   AGE
maindb    1/1       Running   0          2m
web       3/3       Running   0          29s
praparns-MacBook-Pro:multicontainer praparn$ kubectl create -f webmodule_svc.yml
service "web" created
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                     1m
web          10.0.0.69    <nodes>       5000:30661/TCP,80:30500/TCP  4s
praparns-MacBook-Pro:multicontainer praparn$
```

Pods, Container and Service

- Initial database and insert data

```
praparns-MacBook-Pro:multicontainer praparn$ export Server_IP=192.168.99.100
praparns-MacBook-Pro:multicontainer praparn$ export Server_Port=30500
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port
curl: (7) Failed to connect to 192.168.99.100 port 30500: Connection refused
praparns-MacBook-Pro:multicontainer praparn$ clear

praparns-MacBook-Pro:multicontainer praparn$ export Server_IP=192.168.99.101
praparns-MacBook-Pro:multicontainer praparn$ export Server_Port=30500
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 13:18:09 2017
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/init
###########   Database Create New Account Table Done  ###########
```

```
praparns-MacBook-Pro:multicontainer praparn$ curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"
Slave"}' http://$Server_IP:$Server_Port/users/insertuser
HTTP/1.1 200 OK
Server: nginx/1.8.1
Date: Sun, 02 Jul 2017 13:18:45 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive

########### Record was added ###########
praparns-MacBook-Pro:multicontainer praparn$
```

Pods, Container and Service

- Initial database,insert and get data (Direct/Cache)

```
praparns-MacBook-Pro:multicontainer praparn$ export Server_IP=192.168.99.100
praparns-MacBook-Pro:multicontainer praparn$ export Server_Port=30500
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port
curl: (7) Failed to connect to 192.168.99.100 port 30500: Connection refused
praparns-MacBook-Pro:multicontainer praparn$ clear

praparns-MacBook-Pro:multicontainer praparn$ export Server_IP=192.168.99.101
praparns-MacBook-Pro:multicontainer praparn$ export Server_Port=30500
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 13:18:09 2017
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/init
###########   Database Create New Account Table Done  ###########
```

```
praparns-MacBook-Pro:multicontainer praparn$ curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"
Slave"}' http://$Server_IP:$Server_Port/users/insertuser
HTTP/1.1 200 OK
Server: nginx/1.8.1
Date: Sun, 02 Jul 2017 13:18:45 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive

########### Record was added ###########
praparns-MacBook-Pro:multicontainer praparn$
```

```
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Direct)
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Cache)
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/4
Sakkan Yanyicharoen(Database Direct)
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/4
Sakkan Yanyicharoen(Database Cache)
```

Pods, Container and Service

- Delete data (Both cache and database)

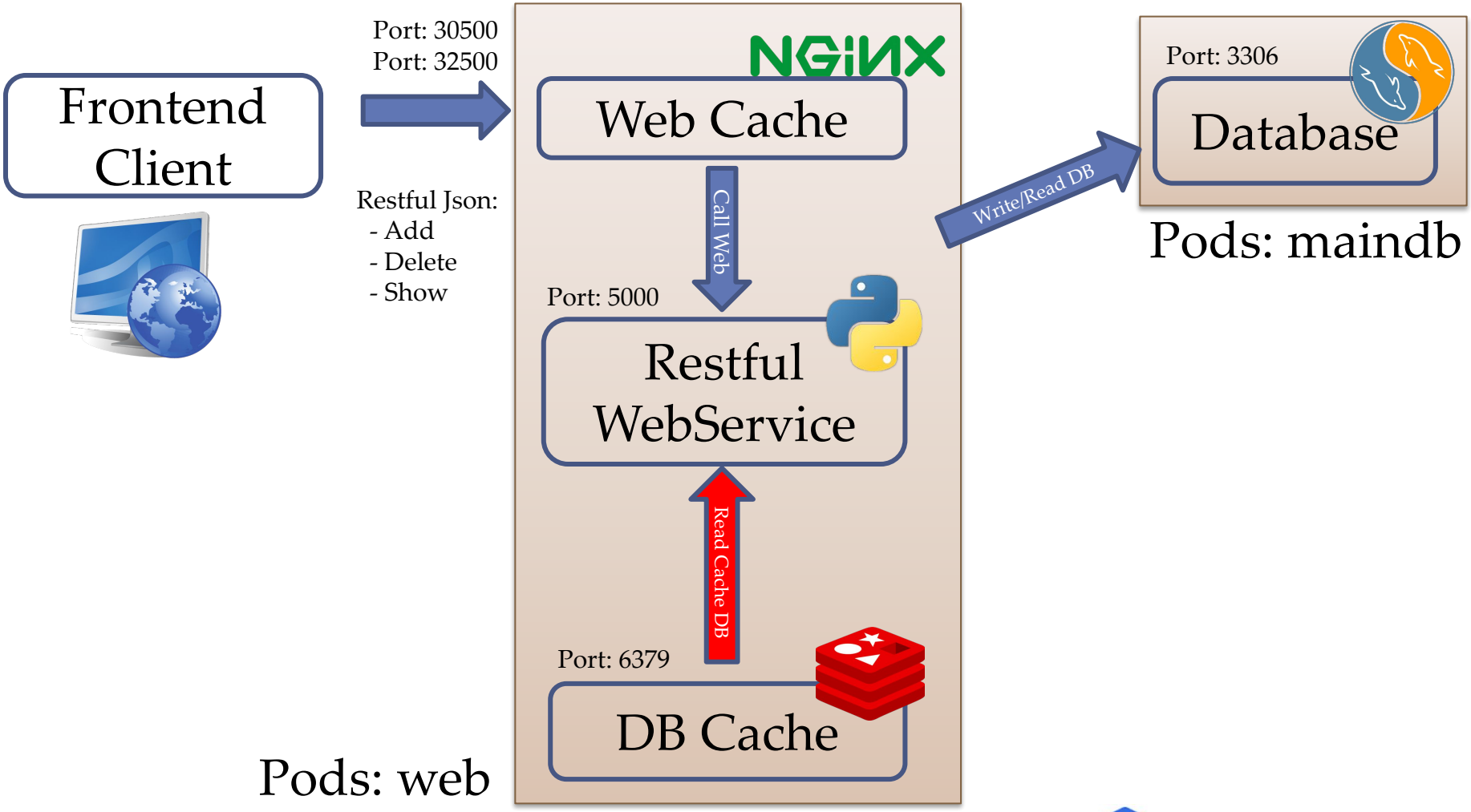
```
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/removeuser/1
########### Record was deleted (Both Database Cache) ###########
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/removeuser/2
########### Record was deleted ###########
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/removeuser/3
########### Record was deleted ###########
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/removeuser/4
########### Record was deleted (Both Database Cache) ###########
praparns-MacBook-Pro:multicontainer praparn$
```

- Remove

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl delete svc maindb web
service "maindb" deleted
service "web" deleted
praparns-MacBook-Pro:multicontainer praparn$ kubectl delete pods maindb web
pod "maindb" deleted
pod "web" deleted
praparns-MacBook-Pro:multicontainer praparn$
```

Workshop: Pods & Service

- Part 2: Deploy web pods with multi container



kubernetes by Google

Pods, Container and Services

- Check how to operate with component on Kubernetes via “Kubectl explain xxx”

kubectl explain <Pods/SVC/etc>

kubectl api-versions

```
ubuntu@ip-10-0-1-239:~$ kubectl explain pods
KIND:     Pod
VERSION:  v1

DESCRIPTION:
     Pod is a collection of containers that can run on a host. This resource is
     created by clients and scheduled onto hosts.

FIELDS:
   apiVersion   <string>
     APIVersion defines the versioned schema of this representation of an
     object. Servers should convert recognized schemas to the latest internal
     value, and may reject unrecognized values. More info:
     https://git.k8s.io/community/contributors/devel/api-conventions.md#resources

   kind <string>
     Kind is a string value representing the REST resource this object
     represents. Servers may infer this from the endpoint the client submits
     requests to. Cannot be updated. In CamelCase. More info:
     https://git.k8s.io/community/contributors/devel/api-conventions.md#types-kinds

   metadata     <Object>
     Standard object's metadata. More info:
     https://git.k8s.io/community/contributors/devel/api-conventions.md#metadata

   spec <Object>
     Specification of the desired behavior of the pod. More info:
     https://git.k8s.io/community/contributors/devel/api-conventions.md#spec-and-status

   status       <Object>
     Most recently observed status of the pod. This data may not be up to date.
     Populated by the system. Read-only. More info:
     https://git.k8s.io/community/contributors/devel/api-conventions.md#spec-and-status

ubuntu@ip-10-0-1-239:~$
```

```
ubuntu@ip-10-0-1-239:~$ kubectl api-versions
admissionregistration.k8s.io/v1beta1
apiextensions.k8s.io/v1beta1
apiregistration.k8s.io/v1
apiregistration.k8s.io/v1beta1
apps/v1
apps/v1beta1
apps/v1beta2
authentication.k8s.io/v1
authentication.k8s.io/v1beta1
authorization.k8s.io/v1
authorization.k8s.io/v1beta1
autoscaling/v1
autoscaling/v2beta1
autoscaling/v2beta2
batch/v1
batch/v1beta1
certificates.k8s.io/v1beta1
coordination.k8s.io/v1
coordination.k8s.io/v1beta1
crd.projectcalico.org/v1
events.k8s.io/v1beta1
extensions/v1beta1
networking.k8s.io/v1
networking.k8s.io/v1beta1
node.k8s.io/v1beta1
policy/v1beta1
rbac.authorization.k8s.io/v1
rbac.authorization.k8s.io/v1beta1
scheduling.k8s.io/v1
scheduling.k8s.io/v1beta1
storage.k8s.io/v1
storage.k8s.io/v1beta1
v1
ubuntu@ip-10-0-1-239:~$
```

Pods, Container and Services

- For check all component on K8S version

kubectl api-resource –o wide

```
ubuntu@ip-10-0-1-239:~$ kubectl api-resources -o wide
NAME                              SHORTNAMES   APIGROUP                       NAMESPACED   KIND                             VERBS
bindings                                                                      true         Binding                          [create]
componentstatuses                 cs                                          false        ComponentStatus                  [get list]
configmaps                        cm                                          true         ConfigMap                        [create delete deletecollection get list patch update watch]
endpoints                         ep                                          true         Endpoints                        [create delete deletecollection get list patch update watch]
events                            ev                                          true         Event                            [create delete deletecollection get list patch update watch]
limitranges                       limits                                      true         LimitRange                       [create delete deletecollection get list patch update watch]
namespaces                        ns                                          false        Namespace                        [create delete get list patch update watch]
nodes                             no                                          false        Node                             [create delete deletecollection get list patch update watch]
persistentvolumeclaims            pvc                                         true         PersistentVolumeClaim            [create delete deletecollection get list patch update watch]
persistentvolumes                 pv                                          false        PersistentVolume                 [create delete deletecollection get list patch update watch]
pods                              po                                          true         Pod                              [create delete deletecollection get list patch update watch]
podtemplates                                                                  true         PodTemplate                      [create delete deletecollection get list patch update watch]
replicationcontrollers            rc                                          true         ReplicationController            [create delete deletecollection get list patch update watch]
resourcequotas                    quota                                       true         ResourceQuota                    [create delete deletecollection get list patch update watch]
secrets                                                                       true         Secret                           [create delete deletecollection get list patch update watch]
serviceaccounts                   sa                                          true         ServiceAccount                   [create delete deletecollection get list patch update watch]
services                          svc                                         true         Service                          [create delete get list patch update watch]
mutatingwebhookconfigurations                  admissionregistration.k8s.io   false        MutatingWebhookConfiguration     [create delete deletecollection get list patch update watch]
validatingwebhookconfigurations                admissionregistration.k8s.io   false        ValidatingWebhookConfiguration   [create delete deletecollection get list patch update watch]
customresourcedefinitions         crd,crds     apiextensions.k8s.io           false        CustomResourceDefinition         [create delete deletecollection get list patch update watch]
apiservices                                    apiregistration.k8s.io         false        APIService                       [create delete deletecollection get list patch update watch]
controllerrevisions                            apps                           true         ControllerRevision               [create delete deletecollection get list patch update watch]
daemonsets                        ds           apps                           true         DaemonSet                        [create delete deletecollection get list patch update watch]
deployments                       deploy       apps                           true         Deployment                       [create delete deletecollection get list patch update watch]
replicasets                       rs           apps                           true         ReplicaSet                       [create delete deletecollection get list patch update watch]
statefulsets                      sts          apps                           true         StatefulSet                      [create delete deletecollection get list patch update watch]
tokenreviews                                   authentication.k8s.io          false        TokenReview                      [create]
localsubjectaccessreviews                      authorization.k8s.io           true         LocalSubjectAccessReview         [create]
selfsubjectaccessreviews                       authorization.k8s.io           false        SelfSubjectAccessReview          [create]
selfsubjectrulesreviews                        authorization.k8s.io           false        SelfSubjectRulesReview           [create]
subjectaccessreviews                           authorization.k8s.io           false        SubjectAccessReview              [create]
horizontalpodautoscalers          hpa          autoscaling                    true         HorizontalPodAutoscaler          [create delete deletecollection get list patch update watch]
cronjobs                          cj           batch                          true         CronJob                          [create delete deletecollection get list patch update watch]
jobs                                           batch                          true         Job                              [create delete deletecollection get list patch update watch]
certificatesigningrequests        csr          certificates.k8s.io            false        CertificateSigningRequest        [create delete deletecollection get list patch update watch]
leases                                         coordination.k8s.io            true         Lease                            [create delete deletecollection get list patch update watch]
bgpconfigurations                              crd.projectcalico.org          false        BGPConfiguration                 [delete deletecollection get list patch create update watch]
bgppeers                                       crd.projectcalico.org          false        BGPPeer                          [delete deletecollection get list patch create update watch]
blockaffinities                                crd.projectcalico.org          false        BlockAffinity                    [delete deletecollection get list patch create update watch]
clusterinformations                            crd.projectcalico.org          false        ClusterInformation               [delete deletecollection get list patch create update watch]
felixconfigurations                            crd.projectcalico.org          false        FelixConfiguration               [delete deletecollection get list patch create update watch]
globalnetworkpolicies                          crd.projectcalico.org          false        GlobalNetworkPolicy              [delete deletecollection get list patch create update watch]
globalnetworksets                              crd.projectcalico.org          false        GlobalNetworkSet                 [delete deletecollection get list patch create update watch]
hostendpoints                                  crd.projectcalico.org          false        HostEndpoint                     [delete deletecollection get list patch create update watch]
ipamblocks                                     crd.projectcalico.org          false        IPAMBlock                        [delete deletecollection get list patch create update watch]
ipamconfigs                                    crd.projectcalico.org          false        IPAMConfig                       [delete deletecollection get list patch create update watch]
ipamhandles                                    crd.projectcalico.org          false        IPAMHandle                       [delete deletecollection get list patch create update watch]
ippools                                        crd.projectcalico.org          false        IPPool                           [delete deletecollection get list patch create update watch]
networkpolicies                                crd.projectcalico.org          true         NetworkPolicy                    [delete deletecollection get list patch create update watch]
networksets                                    crd.projectcalico.org          true         NetworkSet                       [delete deletecollection get list patch create update watch]
events                            ev           events.k8s.io                  true         Event                            [create delete deletecollection get list patch update watch]
daemonsets                        ds           extensions                     true         DaemonSet                        [create delete deletecollection get list patch update watch]
deployments                       deploy       extensions                     true         Deployment                       [create delete deletecollection get list patch update watch]
ingresses                         ing          extensions                     true         Ingress                          [create delete deletecollection get list patch update watch]
networkpolicies                   netpol       extensions                     true         NetworkPolicy                    [create delete deletecollection get list patch update watch]
podsecuritypolicies               psp          extensions                     false        PodSecurityPolicy                [create delete deletecollection get list patch update watch]
replicasets                       rs           extensions                     true         ReplicaSet                       [create delete deletecollection get list patch update watch]
ingresses                         ing          networking.k8s.io              true         Ingress                          [create delete deletecollection get list patch update watch]
```

Pods, Container and Services

- **Init Container**
  - Some scenario. We need to have "predefined" process that need to have some task operate before start main container. So "Init Container" will running specific container before start applications container.
    - Wait other relate component to ready:
      - "for i in {1..100}; do sleep 1; if nslookup myservice; then exit 0; fi; done; exit 1"
    - Prepare configuration:
      - "git clone 'sss.gitbit.com/xx' && sh init.sh"
  - If the "Init Container" is fail. Kubernetes will restart init container until it work.
  - If have multiple init container. Kubernetes will run init container sequential
  - If init container not done. All main container will not start
  - "Init Container"will not support "lifecycle", "livenessProbe", "readinessProbe", or "startupProbe"

Pods, Container and Services

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    app.kubernetes.io/name: MyApp
spec:
  containers:
  - name: myapp-container
    image: busybox:1.28
    command: ['sh', '-c', 'echo The app is running! && sleep 3600']
  initContainers:
  - name: init-myservice
    image: busybox:1.28
    command: ['sh', '-c', "until nslookup myservice.$(cat /var/run/secrets/kubernetes.io/serviceaccount/namespace).svc.cluster.local; do echo waiting for myservice; sleep 2; done"]
  - name: init-mydb
    image: busybox:1.28
    command: ['sh', '-c', "until nslookup mydb.$(cat /var/run/secrets/kubernetes.io/serviceaccount/namespace).svc.cluster.local; do echo waiting for mydb; sleep 2; done"]
```

Pods, Container and Services

- **SideCar Containers (Start active by default on 1.29)**
  - Many use case. We need to have sidecar containers for operate several task along with application container
    - Service Mesh (Kuma, Istio etc.)
    - Monitoring (Dynatrace OneAgent etc.)
    - Security Enhancement
  - SideCar Containers still use under syntax of "Init Container" and define special "restartPolicy:Always"
  - Support "readinessProbe" for defined ready state on pods
  - SideCar also support batch process
  - What is difference from general container (As existing sidecar solution)
    - SideCar run alongside of main container
    - Independence from main container can operate lifecycle,update, patch etc.
    - Still can share resource with main container (Network, Storage etc.)

Pods, Container and Services

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp
  labels:
    app: myapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: myapp
  template:
    metadata:
      labels:
        app: myapp
    spec:
      containers:
        - name: myapp
          image: alpine:latest
          command: ['sh', '-c', 'while true; do echo "logging" >> /opt/logs.txt; sleep 1; done']
          volumeMounts:
            - name: data
              mountPath: /opt
      initContainers:
        - name: logshipper
          image: alpine:latest
          restartPolicy: Always
          command: ['sh', '-c', 'tail -F /opt/logs.txt']
          volumeMounts:
            - name: data
              mountPath: /opt
      volumes:
        - name: data
          emptyDir: {}
```

Pods, Container and Services

- Services
  - Service is independent from Pods
    - Pods is can create / destroy / restart every time (manual/automatic)
    - Mean "Pods" always change their nodes/ip/location everytime
    - Service don't care how Pods is <u>being</u>, But they still can <u>map the Pods</u>

  - Service is abstract of Pods (<u>1 – N Pods</u>)
    - Usually defined port by "Label"
    - Expose access Pods/Load Balance with service (kube-proxy: iptables)
      - "port" that service open for access
      - "target port" that map with Pods for access
      - "type" TCP/UDP
    - Discovery service option:
      - ENVIRONMENT:
        - {SVC_NAME_SERVICE_HOST}
        - {SVC_NAME_SERVICE_PORT}
      - DNS (cluster-addon): Name same service name

Pods, Container and Services

- Services
  - Pods “web”

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: web
  labels:
    name: "web"
    owner: "Praparn_L"
    version: "1.0"
    module: "web"
    environment: "development"
spec:
  containers:
    - name: cachedb
      image: labdocker/redis:latest
      ports:
        - containerPort: 6379
          protocol: TCP
    - name: webservice
      image: labdocker/cluster:webservicev2
      env:
        - name: "REDIS_HOST"
          value: "localhost"
      ports:
        - containerPort: 5000
          protocol: TCP
    - name: webcache
      image: labdocker/cluster:webcache_kubernetes
      ports:
        - containerPort: 80
          protocol: TCP
```

Pods “web2”

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: web2
  labels:
    name: "web"
    owner: "Praparn_L"
    version: "1.0"
    module: "web"
    environment: "development"
spec:
  containers:
    - name: cachedb
      image: labdocker/redis:latest
      ports:
        - containerPort: 6379
          protocol: TCP
    - name: webservice
      image: labdocker/cluster:webservicev2set2
      env:
        - name: "REDIS_HOST"
          value: "localhost"
      ports:
        - containerPort: 5000
          protocol: TCP
    - name: webcache
      image: labdocker/cluster:webcache_kubernetes
      ports:
        - containerPort: 80
          protocol: TCP
```

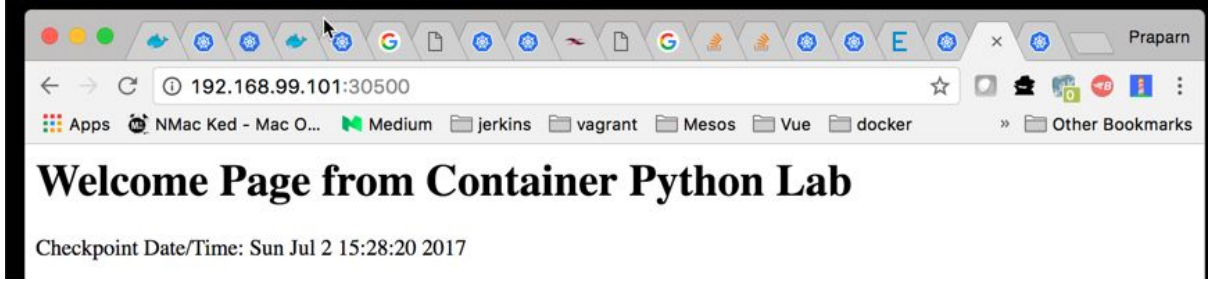
Service “web”

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

Pods, Container and Services

- Services
  - Existing Pods "web"

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

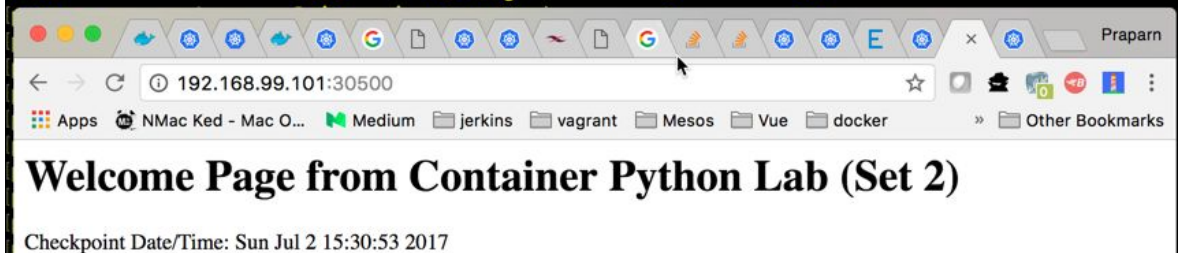



```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)                    AGE
kubernetes   10.0.0.1      <none>        443/TCP                    5d
maindb       10.0.0.236    <none>        3306/TCP                   1m
web          10.0.0.212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

Pods, Container and Services

- Services
  - Replace new Pods "web2"

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




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

Pods, Container and Services

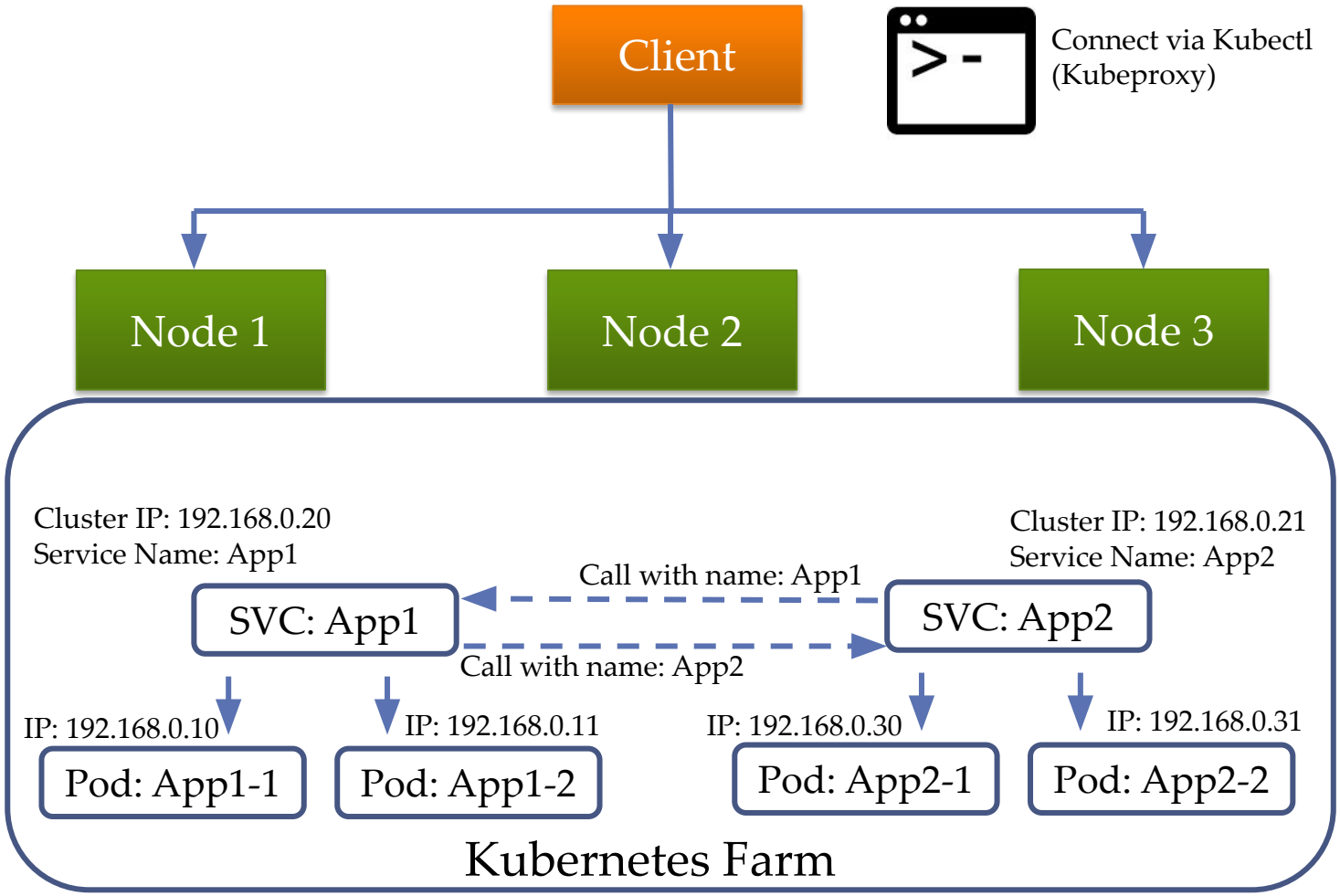
- Services:
  - Publish Service Type:
    - CLUSTER-IP (Default): blind port with ip address of cluster (Accessible from internal cluster system)
    - NodePort: blind port with ip address of node. By default kubernetes will random port (30000-32757). If we need to specify set the option: "nodePort: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

```yaml
  type: NodePort
  ports:
   - port: 5000
     name: webservice
     targetPort: 5000
     protocol: TCP
     nodePort: 32500
   - port: 80
     name: webcache
     targetPort: 80
     protocol: TCP
     nodePort: 30500
```

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

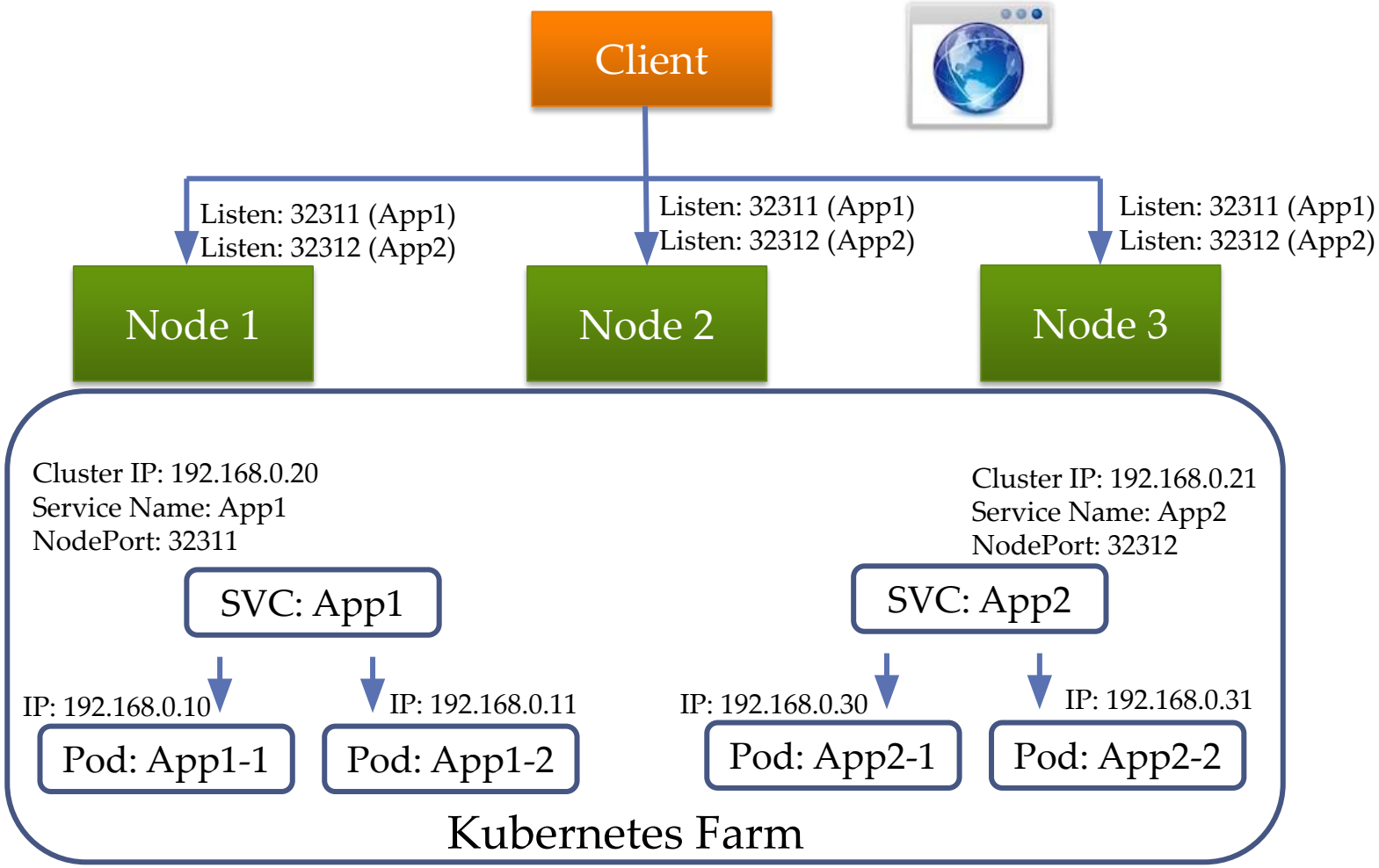
Pods, Container and Services

- Services: ClusterIP



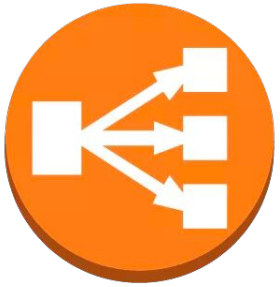
Pods, Container and Services

- Services: NodePort



Pods, Container and Services

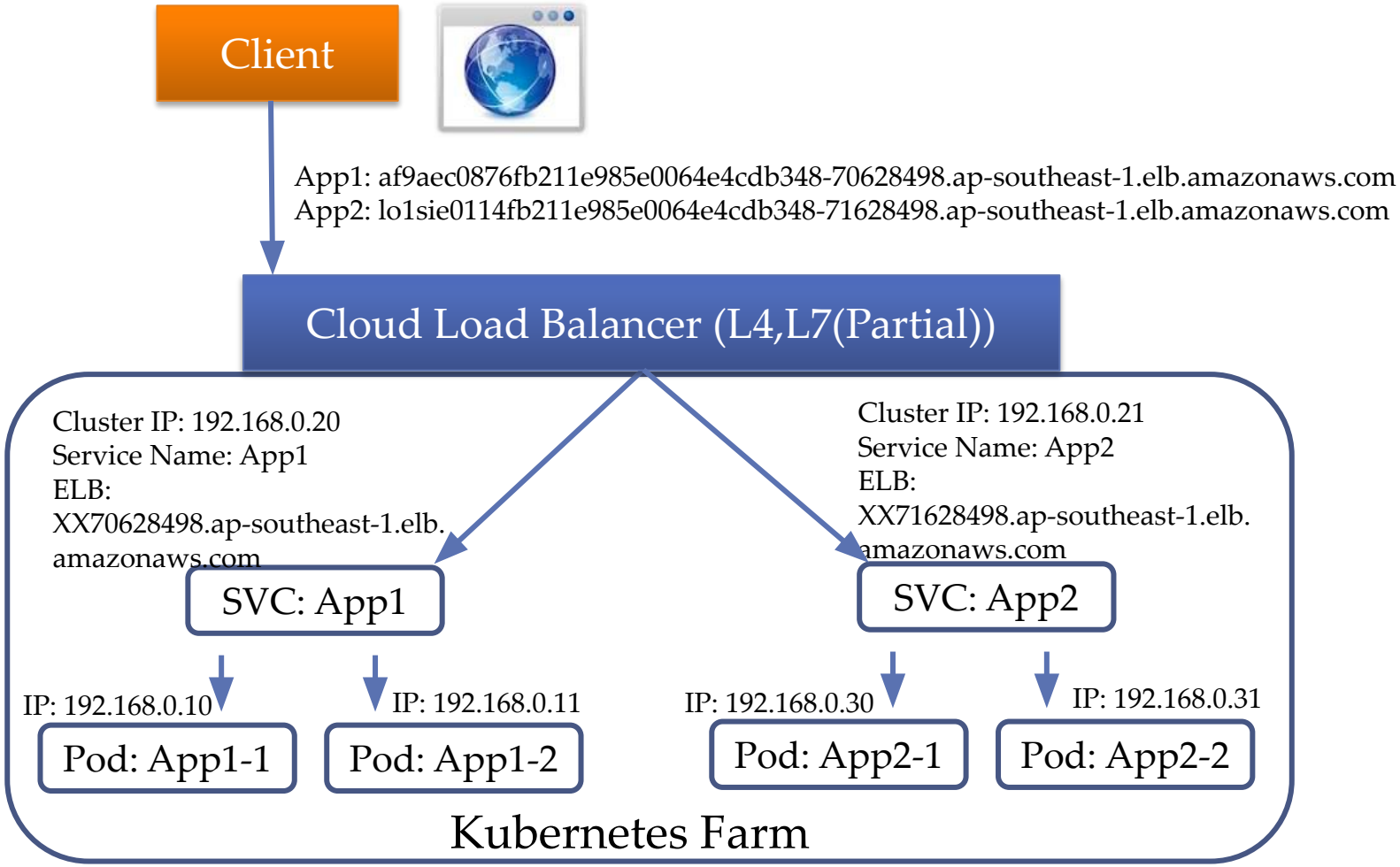
- Services
  - Publish Service Type:
    - LoadBalance:
      - Use load balancer from external cloud provider for intercept traffic
      - Manage by Kubernetes Itself
      - Flexible for use facilities on cloud



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

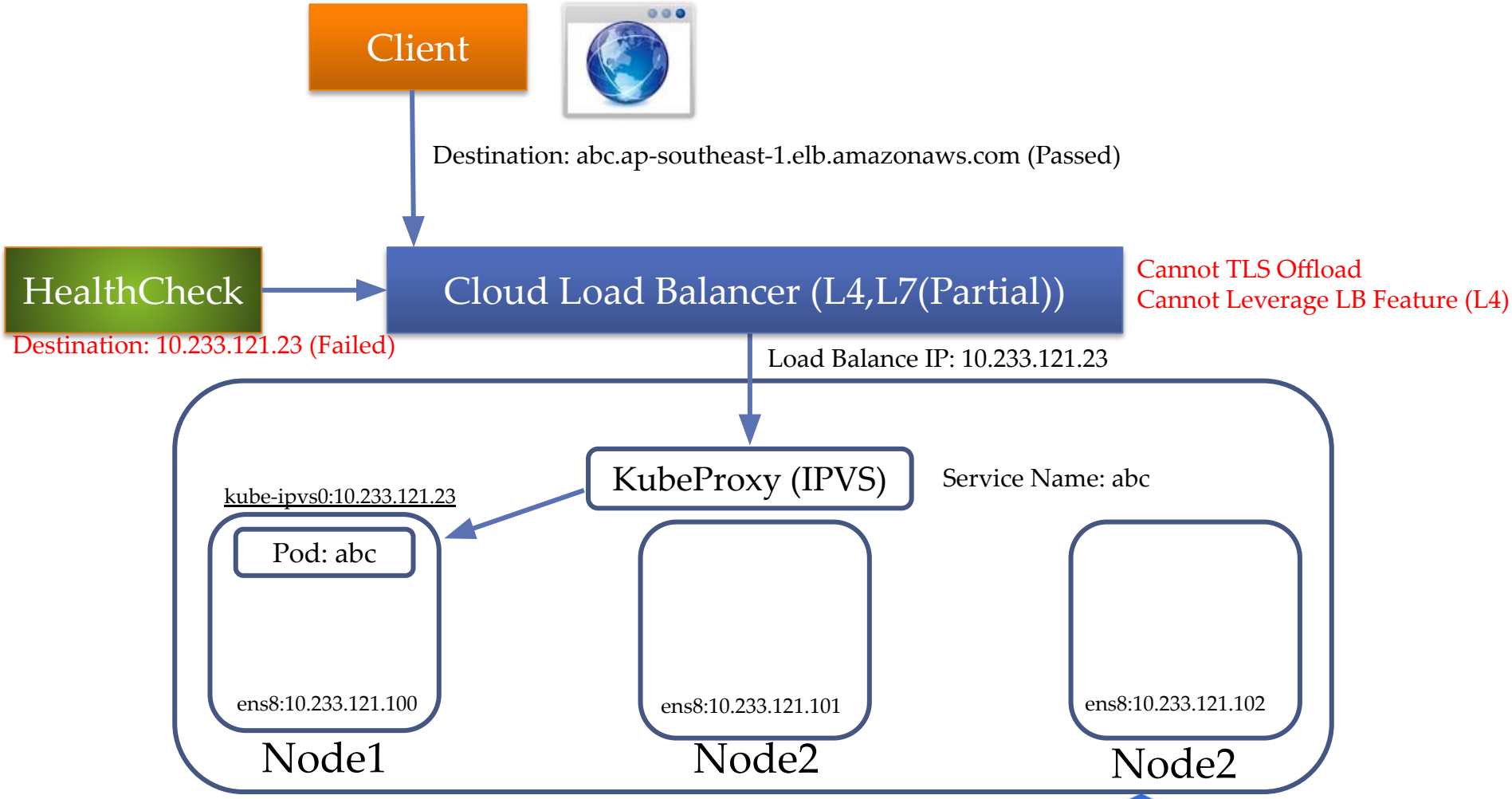
Pods, Container and Services

- Services: Load Balancer



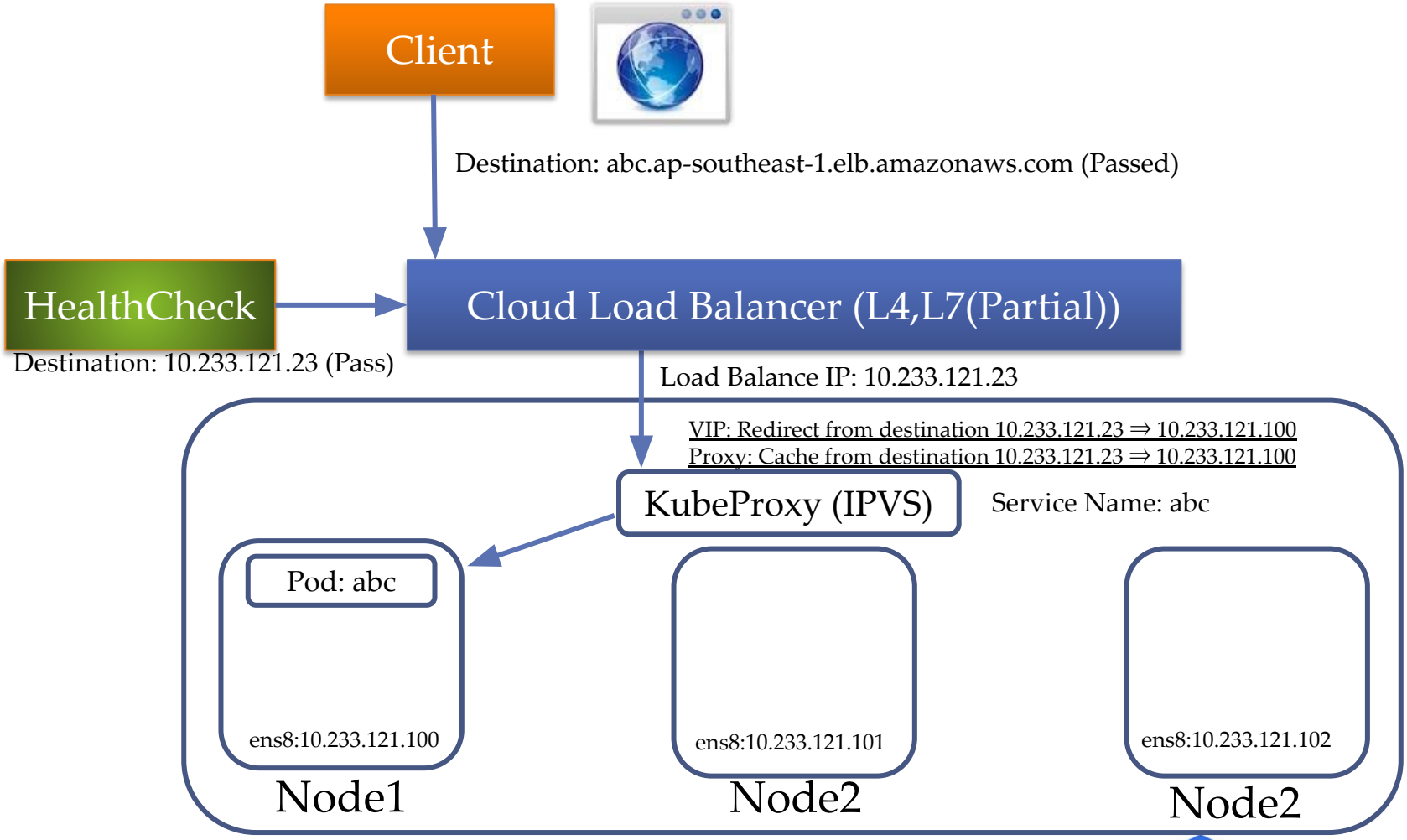
Pods, Container and Services

- Services: Load Balancer (Problem)



Pods, Container and Services

- Services: Load Balancer (ipMode (VIP/Proxy)



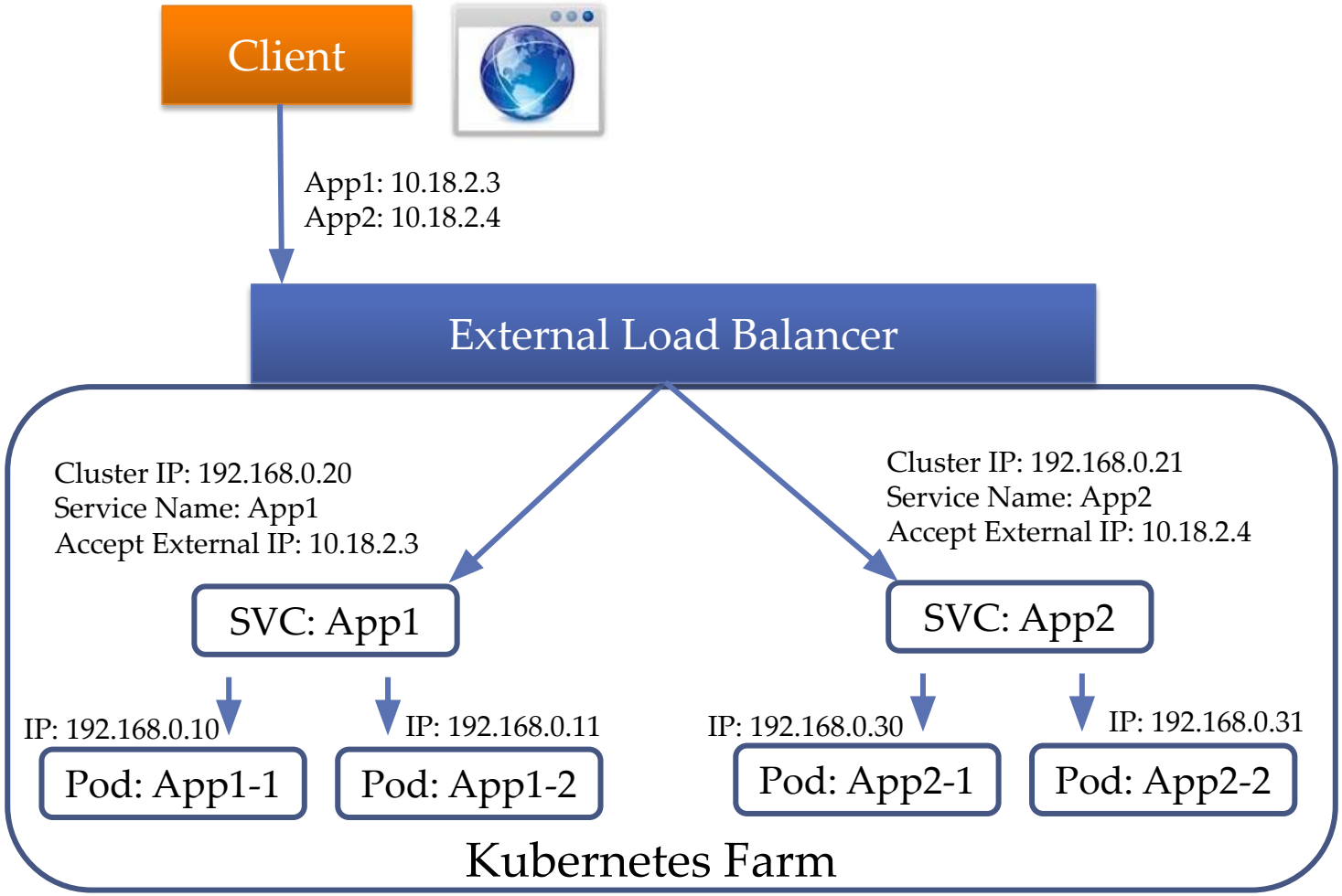
Pods, Container and Services

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

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

Pods, Container and Services

- Services: External IP



Pods, Container and Services

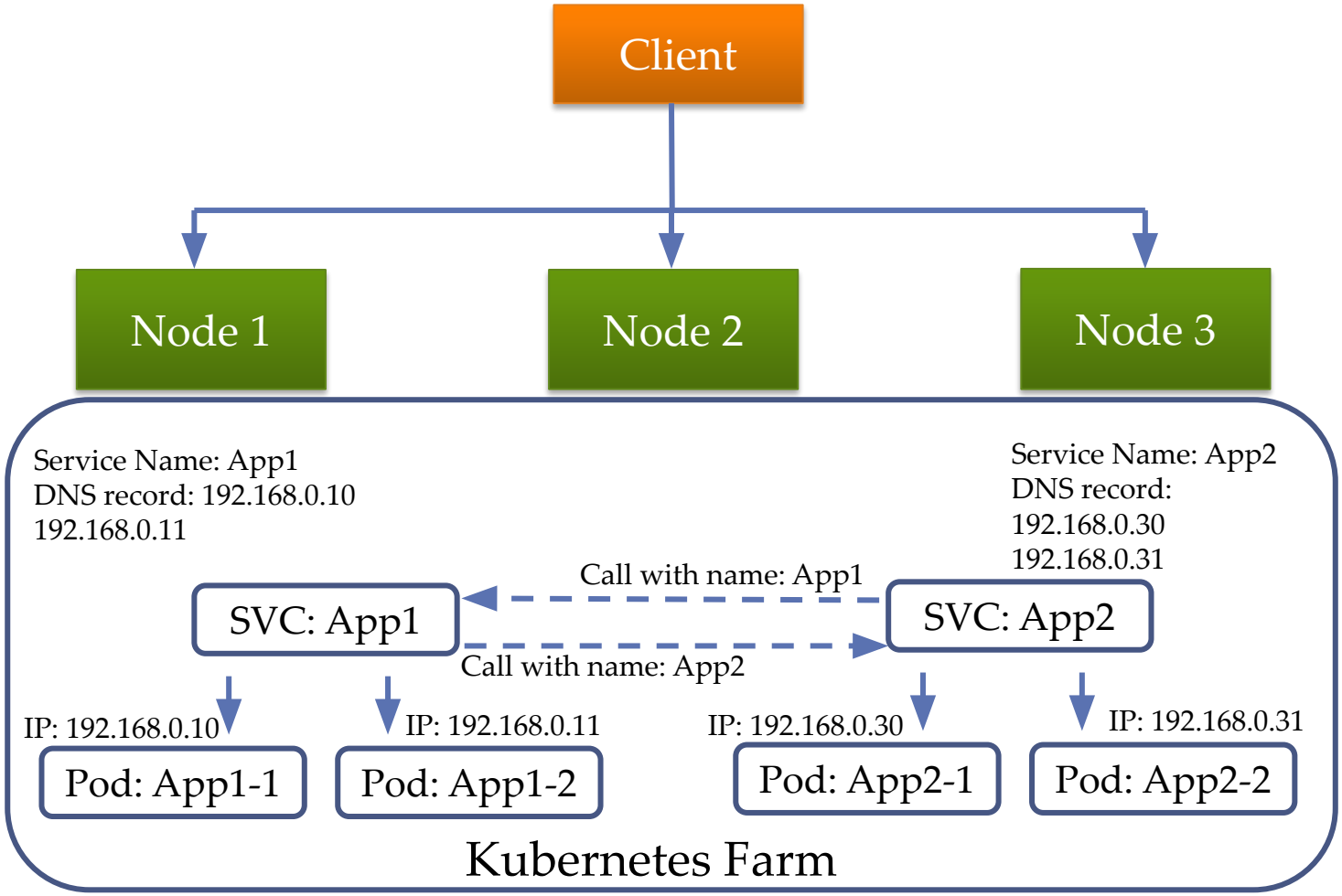
- Services
  - Publish Service Type:
    - Headless Service (StatefulSet): service will just dns round-robin for all statefulset pods's ip address

20dns.yml

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

Pods, Container and Services

- Services: Headless



Pods, Container and Services

- Services
  - Publish Service Type:
    - ExternalName:
      - No selector / No define pods / No endpoints
      - Use for return CNAME record for DNS-Load Balance
      - Work with kube-dns (Kubernetes V1.7 or higher)

```yaml
kind: Service
apiVersion: v1
metadata:
  name: my-service
  namespace: prod
spec:
  type: ExternalName
  externalName: my.database.example.com
```

      - nslookup “my-service.prod.svc.CLUSTER” > return CNAME: “my.database.example.com”

Pods, Container and Services

- Services
  - Without Selector (Not automatic create EndpointSlice)
    - Normally service will be create with automatic endpointslice (Legacy Endpoint)
    - Anyway some use case. We need customize service for special requirement
      - Need to call external redis with service name like proxy mode
      - Need to call service across namespace with simple name service (without fqdn)
      - Need to call across cluster for service
    - With ExternalName type. Kubernetes will not create EndpointSlice by default
      - We need to create EndpointSlice by ourself
      - Customizable for service

Pods, Container and Services

- Services
  - Without Selector

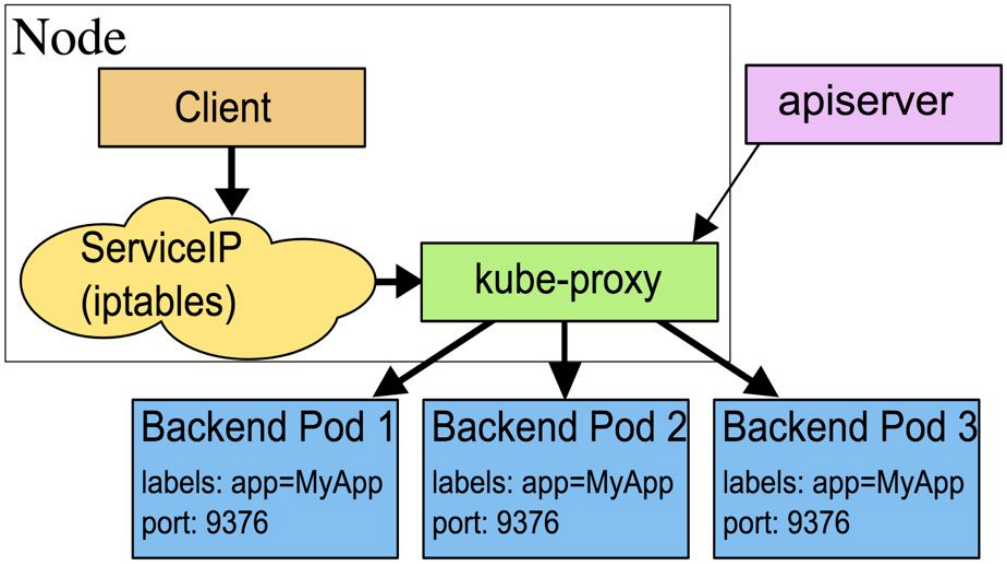
```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  ports:
    - name: http
      protocol: TCP
      port: 80
      targetPort: 9376
```

```yaml
apiVersion: discovery.k8s.io/v1
kind: EndpointSlice
metadata:
  name: my-service-1 # by convention, use the name of the Service
                     # as a prefix for the name of the EndpointSlice
  labels:
    # You should set the "kubernetes.io/service-name" label.
    # Set its value to match the name of the Service
    kubernetes.io/service-name: my-service
addressType: IPv4
ports:
  - name: http # should match with the name of the service port defined above
    appProtocol: http
    protocol: TCP
    port: 9376
endpoints:
  - addresses:
      - "10.4.5.6"
  - addresses:
      - "10.1.2.3"
```

Pods, Container and Services

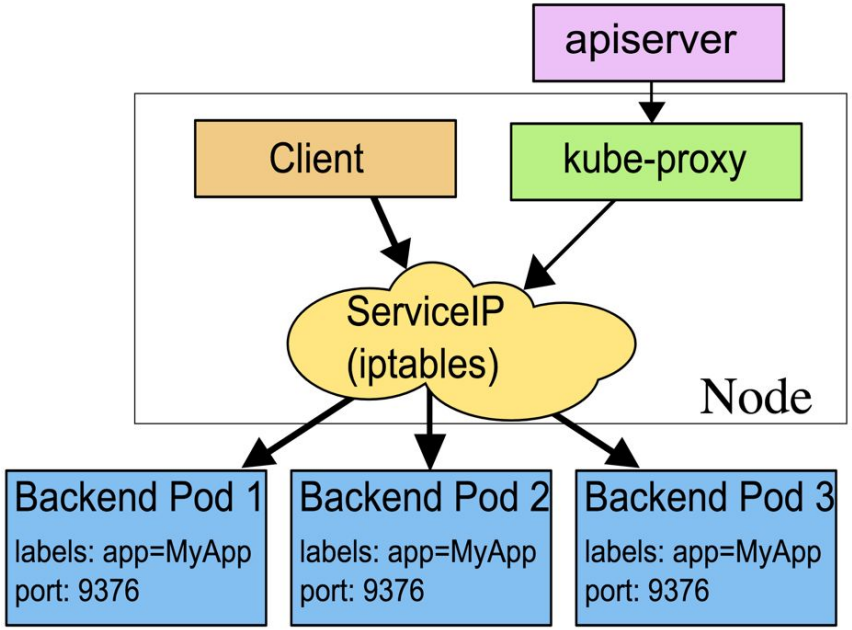
- kube-proxy (proxy-mode)
  - Every node on K8S will use kube-proxy for response to service
    - Initial Virtual IP for service (except External Name)
    - Vary "proxy-mode" for support in kube-proxy

  - Proxy-mode: userspace (default on kubernetes v1.0 – 1.7)
    - Open port on local node (random) and proxy to backend pods
    - Userspace (binary) will terminate and establish new connect to pods
    - Round robin traffic / retry on case pods fail



Pods, Container and Services

- kube-proxy (proxy-mode)
  - **Proxy-mode: iptables (default on kubernetes v1.8 - 1.22)**
    - Install iptable rule on ServiceIp (VIP) for capture traffic and port
    - Redirect traffic to local node for load balance traffic to pods
    - User kernel space that faster than user space (No switch between kernel space and user space)
    - Some distribution was deprecate iptables now !!!



Pods, Container and Services

- kube-proxy (proxy-mode)
  - **Proxy-mode: iptables (default on kubernetes v1.8 - 1.22)**

access.redhat.com/documentation/th-th/red_hat_enterprise_linux/9/html/9.0_release_notes/deprecated_functionality#deprecated-functionality_networking

c Ked - Mac... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | Taiwan | NGINX | Kubernetes | PWA_Progressive... | MYSQL_Cluster | GeneralKB

Expand all | Collapse all

9.0 Release Notes

Making open source more inclusive

Providing feedback on Red Hat documentation

1. Overview

2. Architectures

3. Distribution of content in RHEL 9

4. New features

5. Bug fixes

6. Technology Previews

7. Deprecated functionality

7.1. Installer and image creation

7.2. Security

7.3. Networking

7.3. Networking

ipset and iptables-nft have been deprecated

The `ipset` and `iptables-nft` packages have been deprecated in RHEL. The `iptables-nft` package contains different tools such as `iptables`, `ip6tables`, `ebtables` and `arptables`. These tools will no longer receive new features and using them for new deployments is not recommended. As a replacement, prefer using the `nft` command-line tool provided by the `nftables` package. Existing setups should migrate to `nft` if possible.

When you load the `iptables`, `ip6tables`, `ebtables`, `arptables`, `nft_compat`, or `ipset` module, the module logs the following warning to the `/var/log/messages` file:

```
Warning: <module_name> - this driver is not recommended for new deployment
```

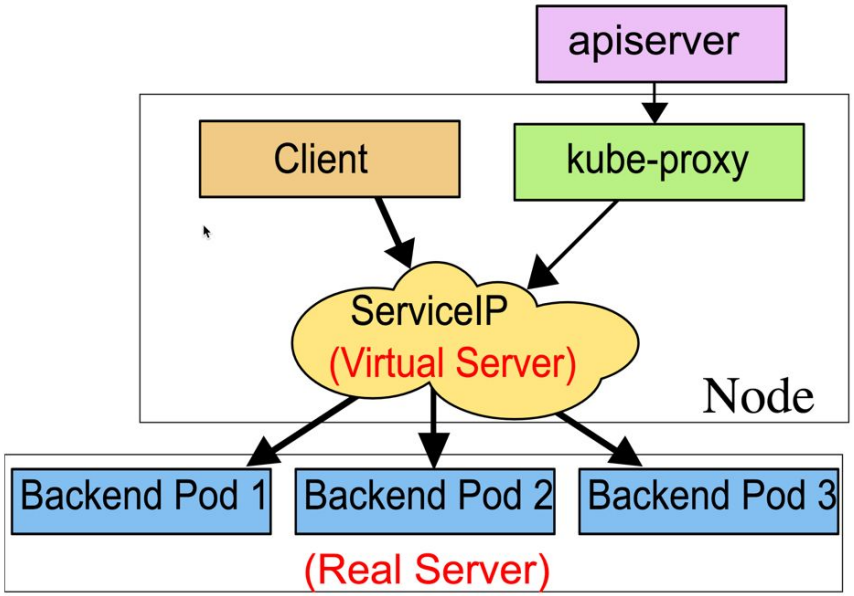
For more information on migrating to nftables, see Migrating from iptables to nftables, as well as the `iptables-translate(8)` and `ip6tables-translate(8)` man pages.

(BZ#1945151)

Ref:
https://access.redhat.com/documentation/th-th/red_hat_enterprise_linux/9/html/9.0_release_notes/deprecated_functionality#deprecated-functionality_networking

Pods, Container and Services

- kube-proxy (proxy-mode)
  - **Proxy-mode: ipvs (GA on kubernetes v1.11)**
    - L4 LB for TCP/UDP with Support Health Check and Retries
    - Create "netlink" interface for establish connection via IPVS
    - Establish IPVS (IP Virtual Server) rule and sync (periodic for consistency before establish)
    - Pure 100% kernel space and better performance
    - Multiple option for load balance
      - rr: round-robin
      - wrr: weight round-robin
      - sh: source hash



Pods, Container and Services

- Configure kube-proxy (proxy-mode)

Run kube-proxy in IPVS mode

Currently, local-up scripts, GCE scripts and kubeadm support switching IPVS proxy mode via exporting environment variables or specifying flags.

Prerequisite

Ensure IPVS required kernel modules (**Notes**: use `nf_conntrack` instead of `nf_conntrack_ipv4` for Linux kernel 4.19 and later)

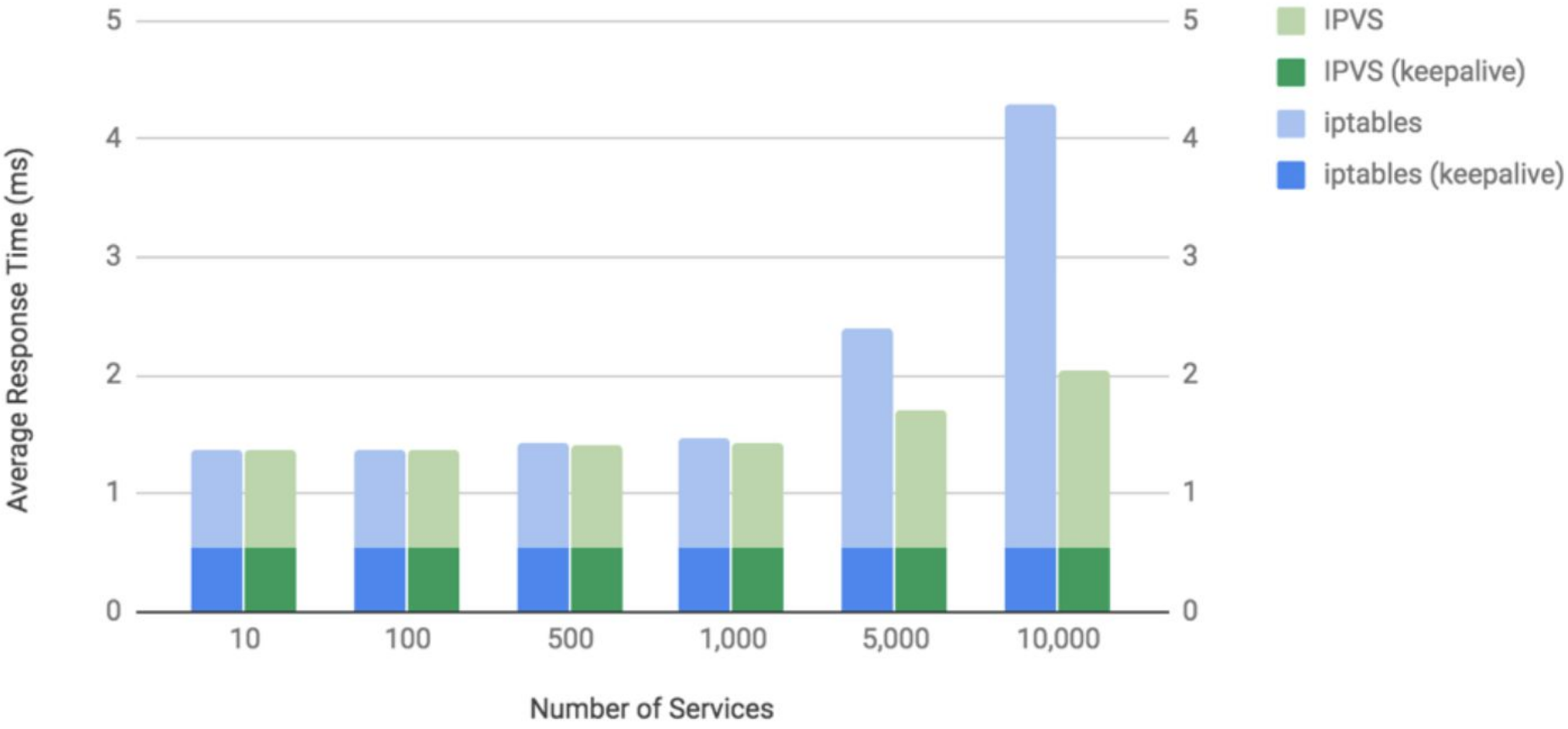
```
ip_vs
ip_vs_rr
ip_vs_wrr
ip_vs_sh
nf_conntrack_ipv4
```

```yaml
---
apiVersion: kubeproxy.config.k8s.io/v1alpha1
kind: KubeProxyConfiguration
mode: ipvs
```

https://github.com/kubernetes/kubernetes/tree/master/pkg/proxy/ipvs

Pods, Container and Services

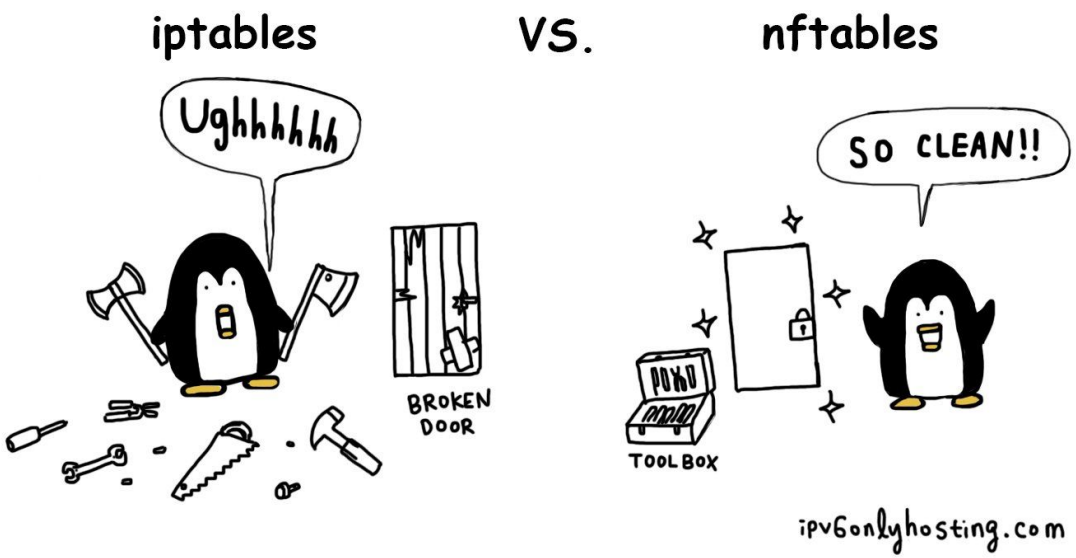
- Compare between IPVS and iptables



Ref: https://www.tigera.io/blog/comparing-kube-proxy-modes-iptables-or-ipvs/

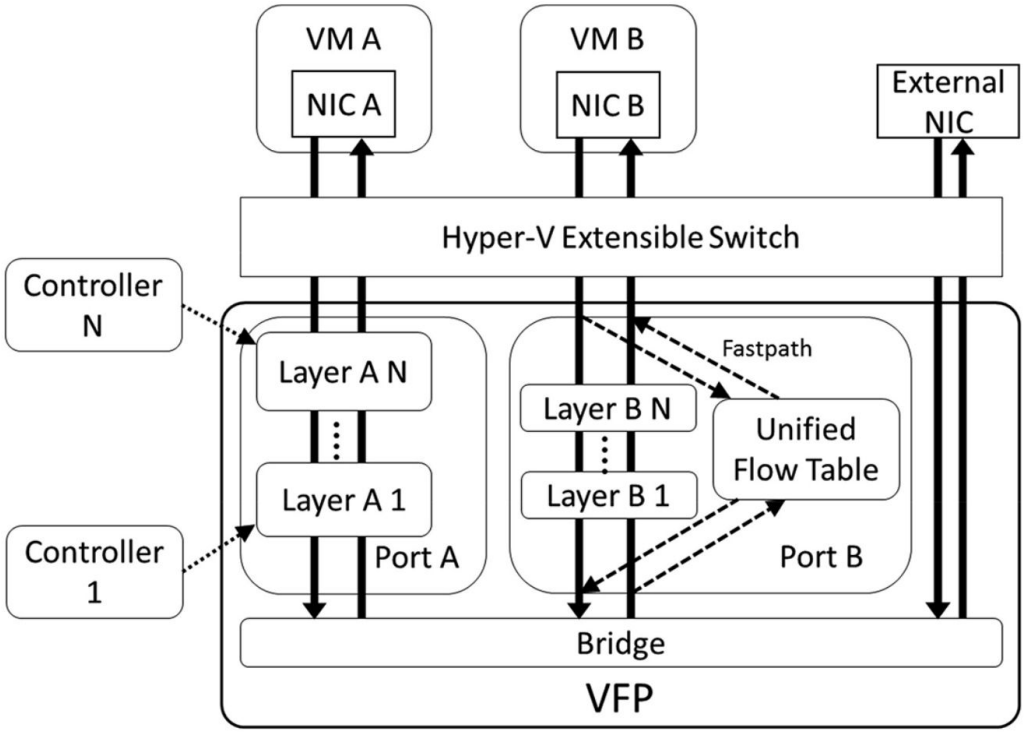
Pods, Container and Services

- kube-proxy (proxy-mode)
  - **Proxy-mode: nftables (GA on kubernetes v1.29)**
    - Replacement with legacy “iptables”
    - nftables is next-generation of iptables with new codebase and improvement (Faster, Support dual stack ipv4/ipv6 etc.)
    - Compatible with existing iptable command on nftables framework
    - Need linux kernel 3.3 and upper
    - Claim to resolve performance issue on iptables



Pods, Container and Services

- kube-proxy (proxy-mode)
  - **Proxy-mode: kernelspace (Start on kubernetes v1.14)**
    - Only available on Windows Node
    - Running on Windows Virtual Filtering Platform (VFP) (Windows VSwitch's extension)



Ref:https://www.usenix.org/system/files/login/articles/login_fall17_02_firestone.pdf

Pods, Container and Services

- Pods Priority and Preemption
  - Give the priority create for pods. When resource is insufficient kubernetes will consider to run higher priority pods first and evict the lower priority pods.
  - Priority will effective only when pods was “created”
  - Disable feature of “Preemption” is not recommend

```yaml
apiVersion: componentconfig/v1alpha1
kind: KubeSchedulerConfiguration
algorithmSource:
  provider: DefaultProvider

...

disablePreemption: true
```

Pods, Container and Services

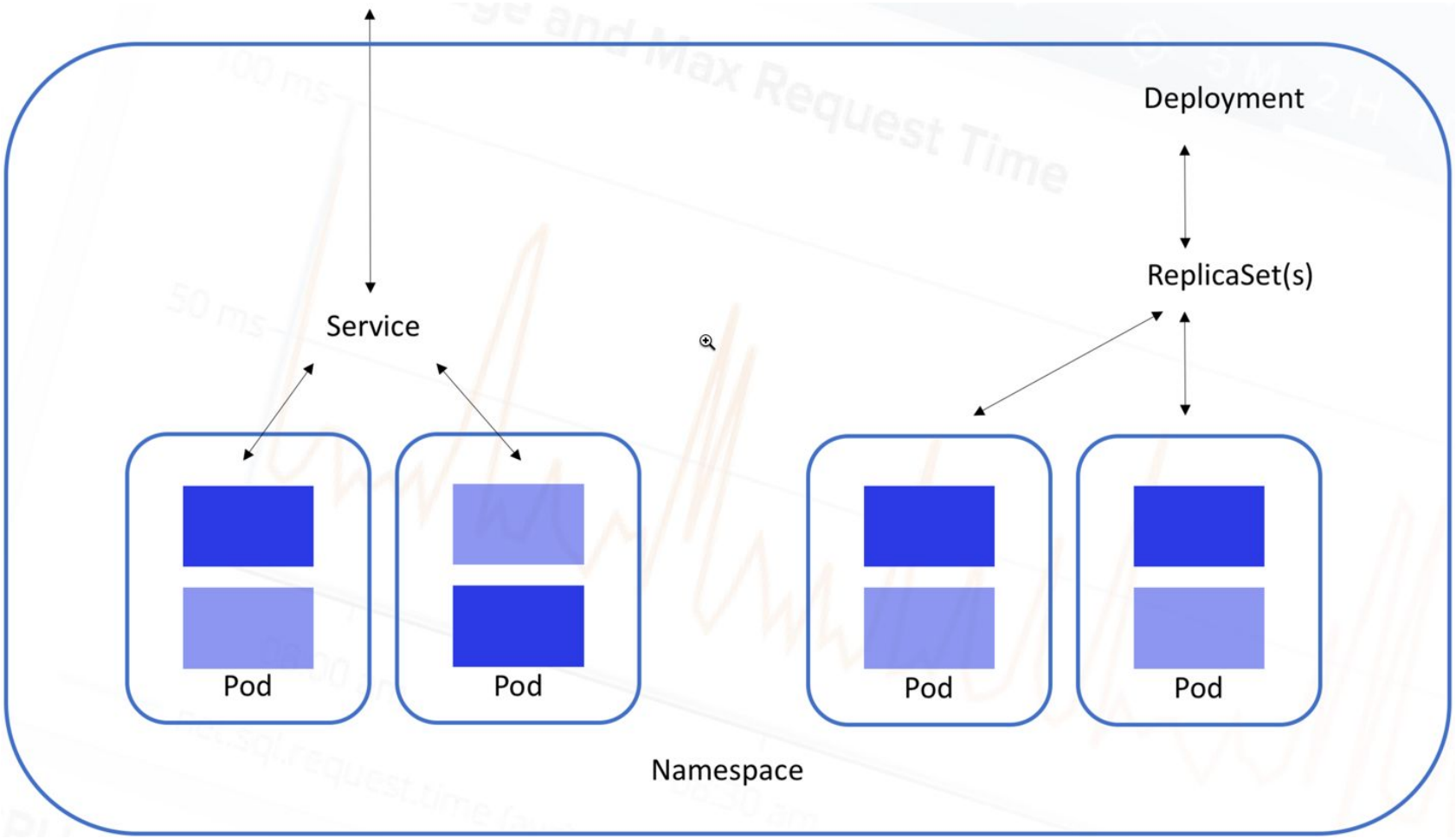
- Pods Priority and Preemption
  - Kubernetes will use value on object “PriorityClass” for consideration resource and **quota**.

```yaml
apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: high-priority
value: 1000000
globalDefault: false
description: "This priority class for highest pods."

---

apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: medium-priority
value: 10000
globalDefault: true
description: "This priority class for default pods."

---

apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: low-priority
value: 100
globalDefault: false
description: "This priority class for lowest 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
  priorityClassName: high-priority
```

Daemon Set and RC

Daemon Set and RC



Daemon Set and RC

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

Replication Controller (RC)

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

Replication Controller (RC)

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

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

Replication Controller (RC)

- Example
  - Pods “webtest”

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

RC “webtest”

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

SVC “webtest”

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

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

Replication Controller (RC)

- Create rc "webtest"

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

- Create svc "webtest"

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

Replication Controller (RC)

- Test delete some Pods from command line

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

- Recheck Pods unit and available

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

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

Replication Controller (RC)

- Scale up replicas on RC

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

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

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

Replication Controller

- Check detail of create process on RC

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

Replication Controller

- 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$
```

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$
```

Workshop: RC

- Deploy replication controller

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

192.168.99.100:32500

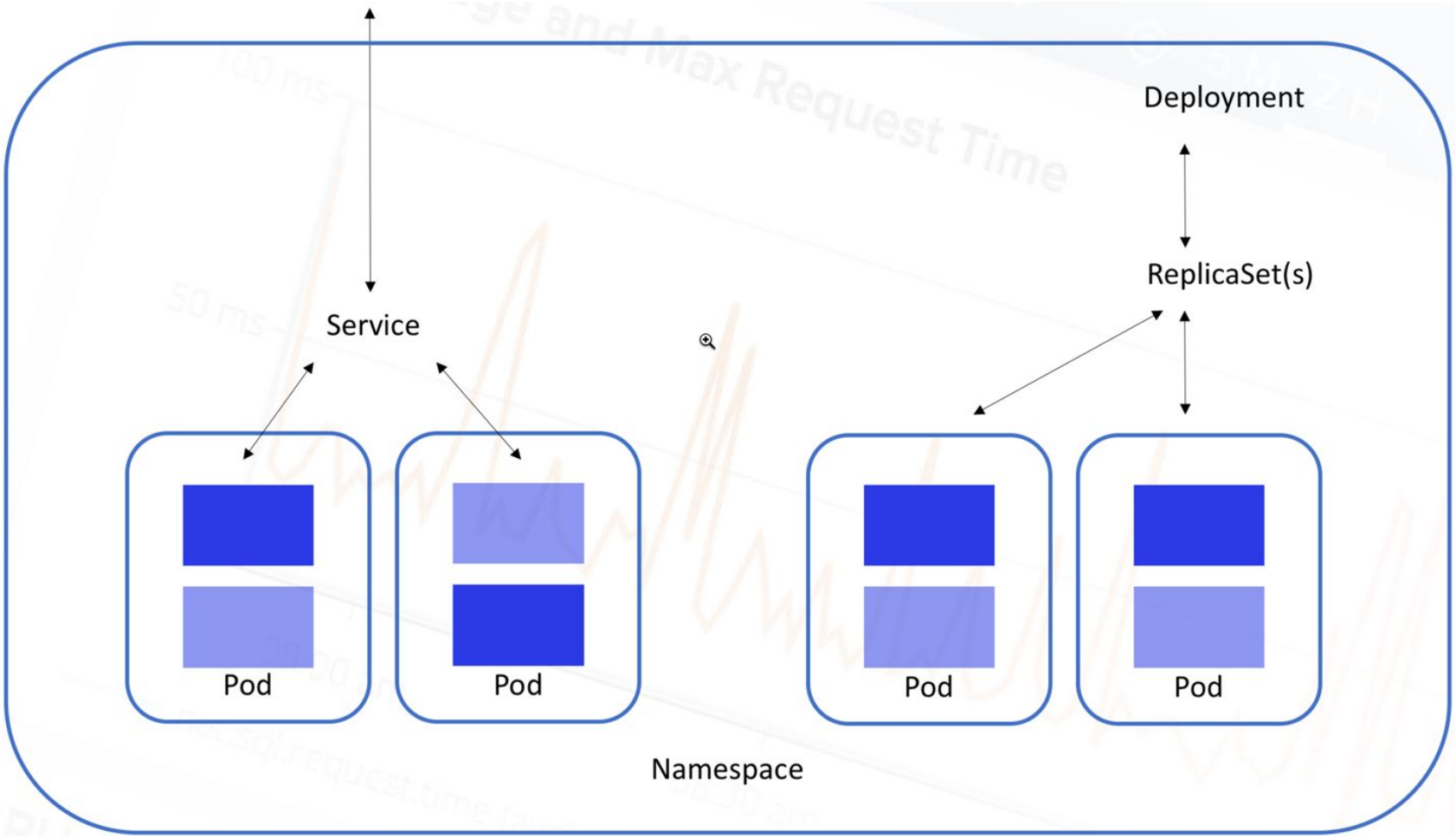
Apps | NMac Ked - Mac O... | Medium | jerkins | vagrant | Mesos | Vue | doc

Welcome Page from Container Python Lab

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

Deployment/RS and Update

Deployment/RS and Update



Deployment/RS and Update

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

Deployment/RS and Update

- ReplicaSet(RS) vs Replication Controller (RC)
  - RS is generation that evolution from RC with capability more dynamic
  - RC support label with method "Equality-based requirement"

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

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

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

Deployment/RS and Update

- Example
  - labdocker/cluster:webservicelite_v1

  - labdocker/cluster:webservicelite_v1.51rc

  - labdocker/cluster:webservicelite_v1.8ga

Deployment/RS and Update

- Example
  - Deployment "webtest"

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

SVC "webtest"

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

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

Deployment/RS and Update

- Create deployment "webtest" and ReplicaSet (RS)

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

- Create svc "webtest"

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

Deployment/RS and Update

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

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

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

Deployment/RS and Update

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

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

  - Step for produce update (rollout) with <u>Each Change (Default)</u>
    - 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 (max Unavailable) and 25% for over design (maxSurge)

Deployment with option --record

Search or jump to... / Pull requests Issues Codespaces Marketplace Explore

kubernetes / kubernetes Public

Watch 3.3k Fork 35.4k Star 96.5k

Code Issues 1.6k Pull requests 794 Actions Projects 6 Security Insights

Deprecate and remove --record flag from kubectl #40422

New issue

Open soltysh opened this issue on Jan 25, 2017 · 55 comments

soltysh commented on Jan 25, 2017 Contributor

Currently `--record` flag seems like a really bad decision made some time ago, and supporting it makes live harder when trying to modify any parts of code (see this discussion). I'm proposing to deprecate that flag and drop it entirely in future version.

@liggitt fyi
@kubernetes/sig-cli-feature-requests opinions?

8 2

soltysh added area/kubectl sig/cli sig/windows labels on Jan 25, 2017

soltysh mentioned this issue on Jan 25, 2017

make kubectl generic commands work with unstructured objects #40260 Merged

19 tasks

Assignees
No one assigned

Labels
area/kubectl kind/feature lifecycle/frozen sig/cli

Projects
None yet

Milestone
No milestone

Development
No branches or pull requests

Ref: https://github.com/kubernetes/kubernetes/issues/40422#issuecomment-1426936343

kubernetes by Google

Deployment with option --record

praparn commented on Sep 11, 2021

Is it possible for append this option on deployment yaml itself? A lot of need to use this feature for check and record history purpose.

pacoxu mentioned this issue on Sep 16, 2021

kubectl --record should avoid capturing secrets on the command line #20508 — Closed

sftim mentioned this issue on Oct 6, 2021

Add change-cause annotation to labels/annotations page kubernetes/website#29896 — Merged

eddiezane mentioned this issue on Oct 29, 2021

`set image deployment` **without** `--record` **re-uses stale command** kubernetes/kubectl#1137 — Closed

Ref: https://github.com/kubernetes/kubernetes/issues/40422#issuecomment-1426936343

Deployment with option --record

ric79 commented on Aug 2, 2022

Here my example

```
$ k version --short
Client Version: v1.24.3
Kustomize Version: v4.5.4
Server Version: v1.24.0

$ k create deployment nginx-dep --image=nginx:1.22.0-alpine-perl --replicas 5

$ k rollout history deployment nginx-dep
deployment.apps/nginx-dep
REVISION  CHANGE-CAUSE
1         <none>

$ k set image deployment nginx-dep nginx=nginx:1.23-alpine-perl
$ k annotate deployment nginx-dep kubernetes.io/change-cause="demo version changed from 1.22.0 to 1.23.0" --ov
$ k rollout history deployment nginx-dep
deployment.apps/nginx-dep
REVISION  CHANGE-CAUSE
1         <none>
2         demo version changed from 1.22.0 to 1.23.0

$ k set image deployment nginx-dep nginx=nginx:1.23.1-alpine-perl
$ k annotate deployment nginx-dep kubernetes.io/change-cause="demo version changed from 1.23.0 to 1.23.1" --ov
$ k rollout history deployment nginx-dep
deployment.apps/nginx-dep
REVISION  CHANGE-CAUSE
1         <none>
2         demo version changed from 1.22.0 to 1.23.0
3         demo version changed from 1.23.0 to 1.23.1
```

gjkim42 mentioned this issue on Aug 7, 2022

"kubectl rollout history" produces wrong entry when --record not explictly used in "set

Closed

Ref: https://github.com/kubernetes/kubernetes/issues/40422#issuecomment-1426936343

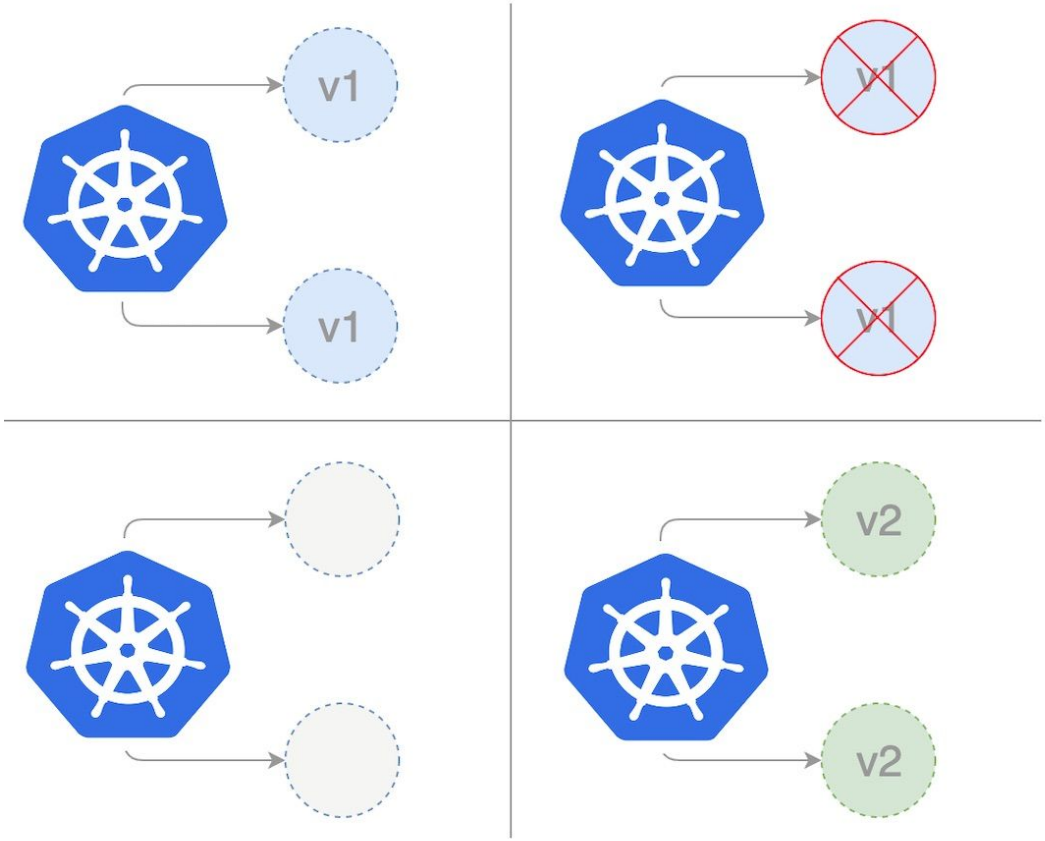
Deployment/RS and Update

- Let's Talk about deployment 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

Ref:
https://container-solutions.com/kubernetes-deployment-strategies/?fbclid=IwAR1T3mdkK46HmJ7ckla6iEjtAuKKnijLWRcsrK3Jetzb5BOB21akiYBgm5Y
https://auth0.com/blog/deployment-strategies-in-kubernetes/

Deployment/RS and Update

- Recreate

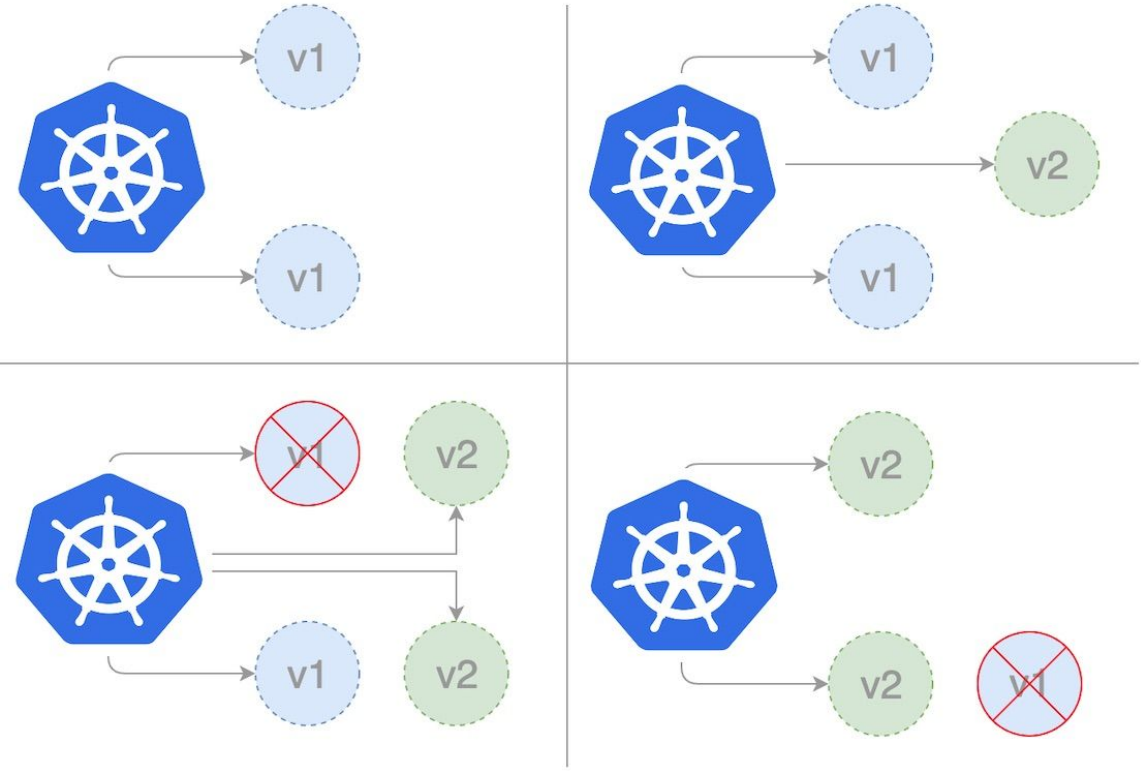


webtest_deploy.yml

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

Deployment/RS and Update

- RollingUpdate (Default)



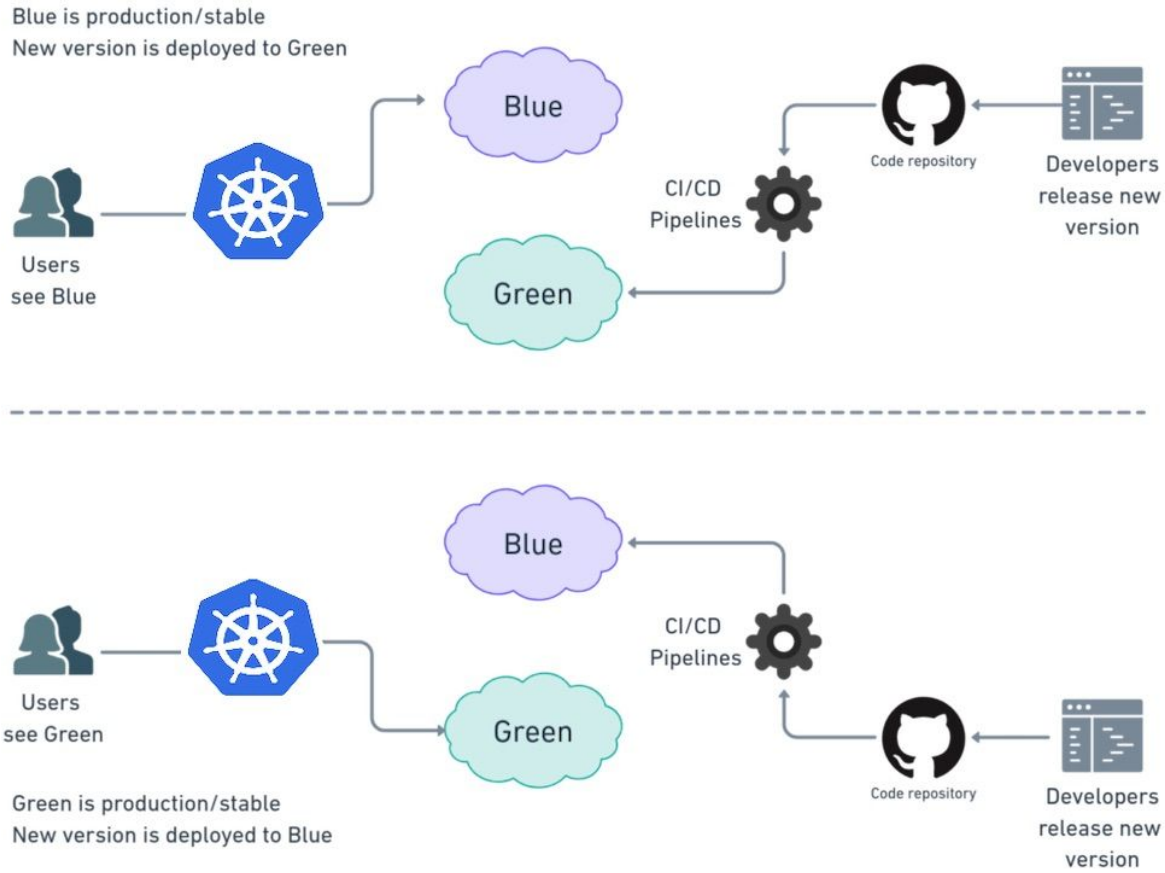
webtest_deploy.yml

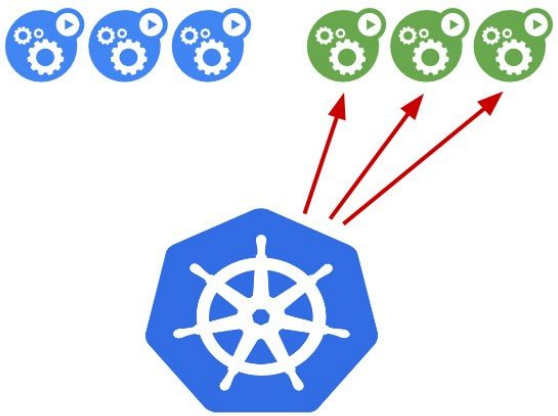
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 4
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavaible: 1
      maxSurge: 1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

kubernetes by Google

Deployment/RS and Update

- Blue/Green



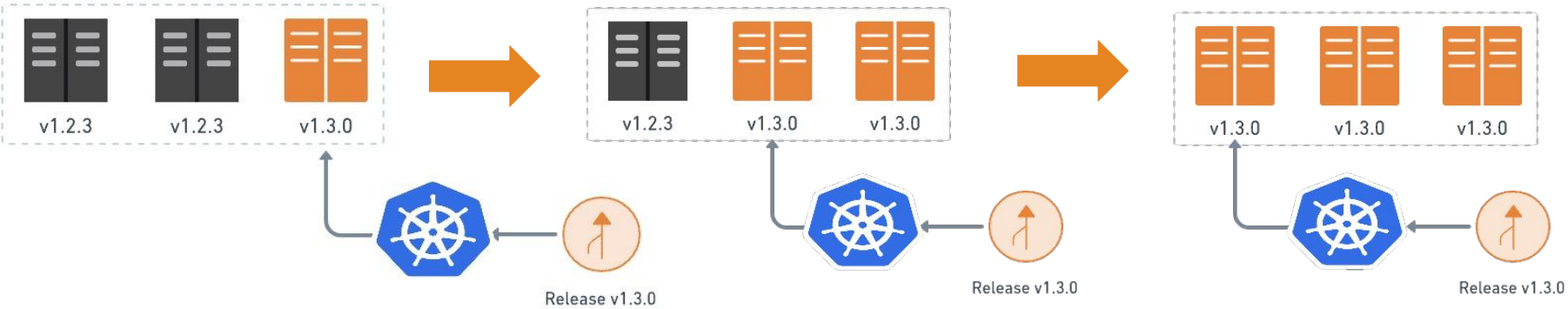


webtest_deploy.yml webtest_svc.yml

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

Deployment/RS and Update

- Canary: (2 Deployment same label)




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

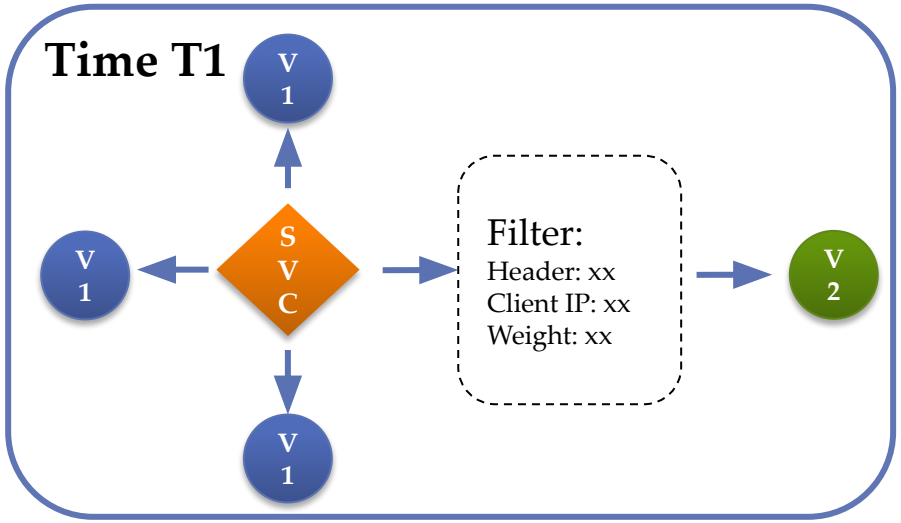
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest_V2
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
```

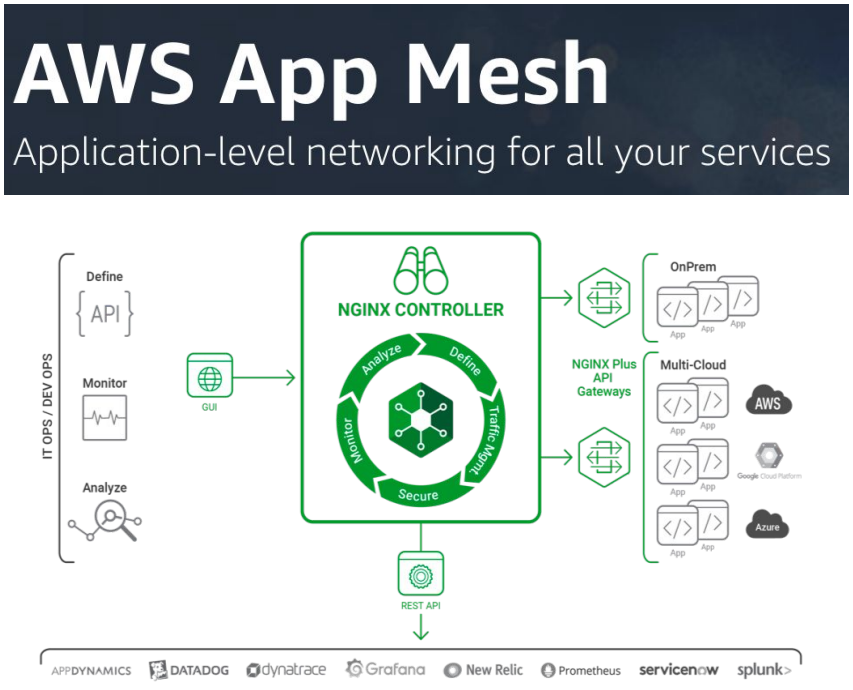
webtest_deploy.yml ● webtest_svc.yml ×

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

Deployment/RS and Update

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





LINKERD

Istio

Connect, secure, control, and observe services.

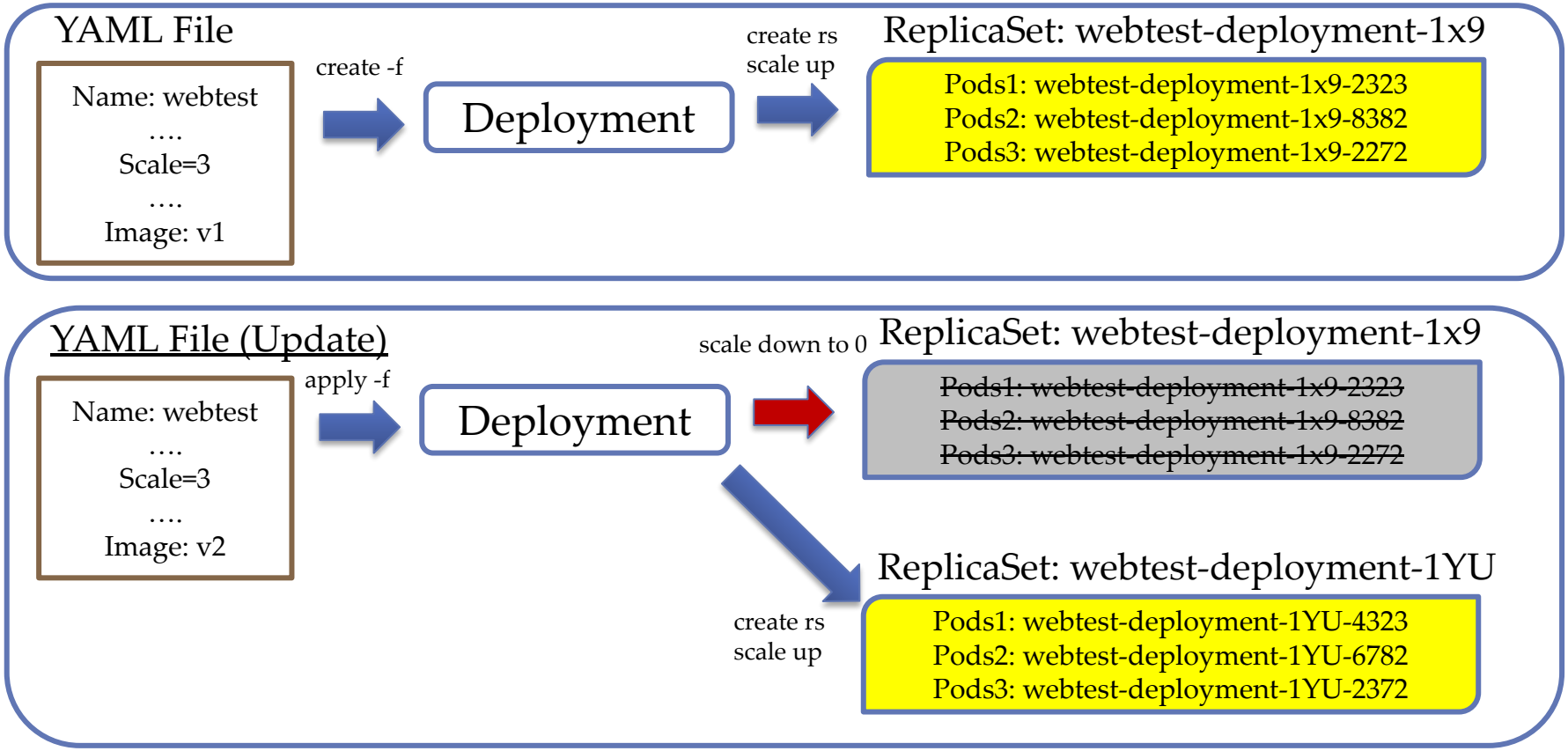
vamp

Kubernetes: Production Workload Orchestration

kubernetes by Google

Deployment/RS and Update

- 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

- Define "revisionHistoryLimit" as design (Default: 10)

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

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>
```

Deployment/RS and Update

- Check status of deployment's update (rollout)
  - Use "rollout" utility (status/history/pause/resume/undo)

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-10-200-20-82:~$ kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.51rc --record=true
        kubectl annotate deployment webtest kubernetes.io/change-cause="Second Release version 1.5" --overwrite=true
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/webtest image updated
deployment.apps/webtest annotated
ubuntu@ip-10-200-20-82:~$ kubectl rollout status deployment/webtest
Waiting for deployment "webtest" rollout to finish: 1 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 1 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 1 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
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-10-200-20-82:~$
```

Deployment/RS and Update

- Deployment update strategy
  - Set new image on deployment

```
ubuntu@ip-10-200-20-82:~$ kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.8ga --record=true
        kubectl annotate deployment webtest kubernetes.io/change-cause="Third Release version 1.8ga" --overwrite=true
Flag --record has been deprecated, --record will be removed in the future
deployment.apps/webtest image updated
deployment.apps/webtest annotated
ubuntu@ip-10-200-20-82:~$ kubectl rollout pause deployment/webtest
deployment.apps/webtest paused
ubuntu@ip-10-200-20-82:~$ kubectl get rs
NAME                 DESIRED   CURRENT   READY   AGE
webtest-78bc7494f5   3         3         3       63s
webtest-7c695d86fd   1         1         1       11s
webtest-9d9bcbc6d    0         0         0       4m14s
ubuntu@ip-10-200-20-82:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
webtest-78bc7494f5-6hmq5   1/1     Running   0          60s
webtest-78bc7494f5-b8zgq   1/1     Running   0          67s
webtest-78bc7494f5-w2zlr   1/1     Running   0          58s
webtest-7c695d86fd-skbm8   1/1     Running   0          15s
ubuntu@ip-10-200-20-82:~$ kubectl rollout resume deployment/webtest
deployment.apps/webtest resumed
ubuntu@ip-10-200-20-82:~$ kubectl rollout status deployment/webtest
deployment "webtest" successfully rolled out
ubuntu@ip-10-200-20-82:~$
```

Deployment/RS and Update

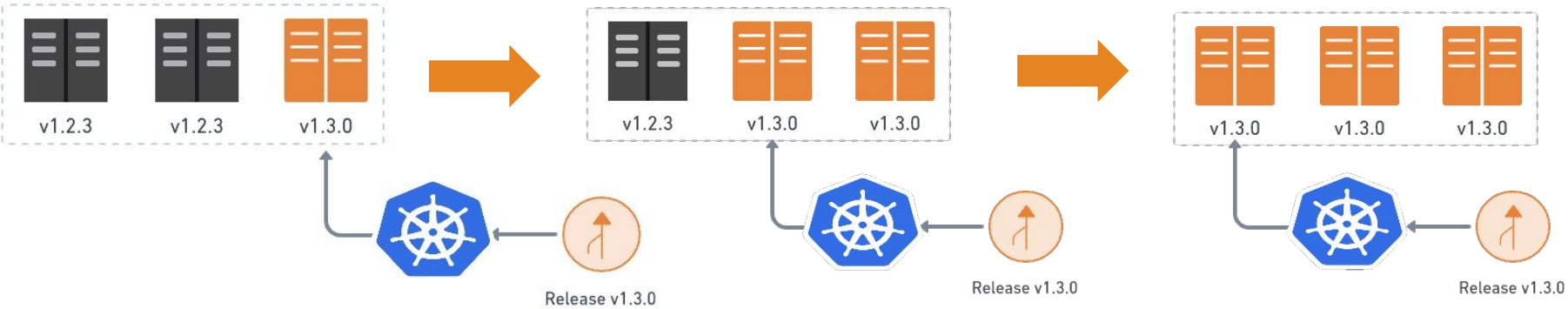
- Deployment update strategy
  - Rollout History / Rollback

```
ubuntu@ip-10-200-20-82:~$ kubectl rollout history deployment/webtest
deployment.apps/webtest
REVISION  CHANGE-CAUSE
1         First Release version 1.0
2         Second Release version 1.5
3         Third Release version 1.8ga

ubuntu@ip-10-200-20-82:~$
ubuntu@ip-10-200-20-82:~$ kubectl rollout undo deployment/webtest --to-revision=2
deployment.apps/webtest rolled back
ubuntu@ip-10-200-20-82:~$ kubectl rollout status deployment/webtest
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 old replicas are pending termination...
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-10-200-20-82:~$
```

Deployment/RS and Update

```
ubuntu@ip-10-200-20-82:~$ kubectl rollout history deployment/webtest
deployment.apps/webtest
REVISION  CHANGE-CAUSE
1         First Release version 1.0
3         Third Release version 1.8ga
4         Second Release version 1.5

ubuntu@ip-10-200-20-82:~$ kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1
        kubectl annotate deployment webtest kubernetes.io/change-cause="Fourth Release version 1.0" --overwrite=true
        kubectl rollout status deployment/webtest
        kubectl rollout history deployment/webtest
deployment.apps/webtest image updated
deployment.apps/webtest annotated
Waiting for deployment "webtest" rollout to finish: 1 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 1 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 1 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
Waiting for deployment "webtest" rollout to finish: 2 out of 3 new replicas have been updated...
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
deployment.apps/webtest
REVISION  CHANGE-CAUSE
3         Third Release version 1.8ga
4         Second Release version 1.5
5         Fourth Release version 1.0

ubuntu@ip-10-200-20-82:~$ kubectl get rs
NAME                 DESIRED   CURRENT   READY   AGE
webtest-78bc7494f5   0         0         0       2m26s
webtest-7c695d86fd   0         0         0       106s
webtest-9d9bcbc6d    3         3         3       2m49s
ubuntu@ip-10-200-20-82:~$
```

Workshop: Deployment




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

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

webtest_deploy.yml ● | webtest_svc.yml ×

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

Volume

Volume

- By default all container in same Pods will share storage (emptyDir)
  - This storage will keep all read/write for all container in Pods
  - Persistent storage for container as long as Pods still avaliable
  - Container crash not effect to this storage type
  - When Pods was delete from node. emptyDir was deleted forever
- Data in container/Pods is “ephemeral” that may loss when Pods was restart/distribute
- Kubernetes support many type of volume on system

Volume

- List of support storage
  - EmptyDir
  - **hostPath**
  - local
  - Nfs
  - Iscsi
  - fc(fibre channel)
  - downwardAPI
  - Rbd
  - Cephfs (deprecated)
  - gitRepo (deprecated)
  - Secret
  - ***persistentVolumeClaim (PVC) (Talk Day2)***
  - cinder (deprecated)
  - awsElasticBlockStore (deprecated)
  - azureDisk (deprecated)
  - azureFile (deprecated)
  - gcePersistentDisk (remove 1.29)
  - gitRepo (deprecated)
  - portworxVolume (deprecated)
  - vsphereVolume (deprecated)
  - flexVolume (deprecated)
  - ConfigMap
  - Secret

Volume

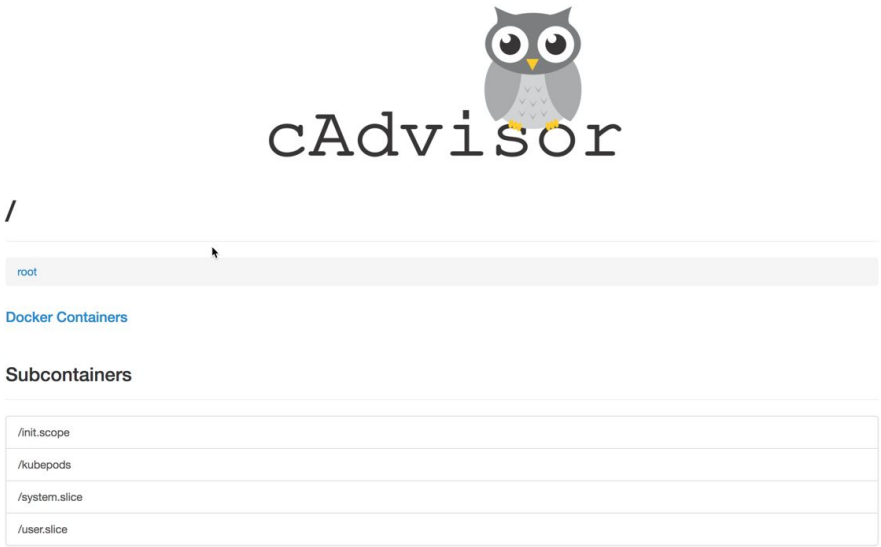
- hostPath:
  - Mount file/directory on host to Pods
  - Recommend for Pods that need read some data on host path (Such as cAdvisor etc)
  - Consideration
    - hostPath may effect to some of path/file not equal on different host
    - Kubernetes resource scheduling cannot check readiness of hostPath
    - Some hostPath may need privilege for access. This need to consideration.

Volume

- hostPath:

Users > praparnlueangphoonlap > Work > Kubernetes > Train

```yaml
    labels:
      name: cadvisor
      owner: Praparn_L
      version: "1.0"
      module: Monitor
      environment: development
  spec:
    replicas: 1
    selector:
      matchLabels:
        name: cadvisor
        owner: Praparn_L
        version: "1.0"
        module: Monitor
        environment: development
    template:
      metadata:
        labels:
          name: cadvisor
          owner: Praparn_L
          version: "1.0"
          module: Monitor
          environment: development
      spec:
        containers:
          - name: cadvisor
            volumeMounts:
            - mountPath: /var/run
              name: volrun
            - mountPath: /sys
              name: volsys
            - mountPath: /var/lib/containerd/
              name: voldocker
            image: labdocker/cadvisor:latest
            ports:
            - containerPort: 8080
              protocol: TCP
        volumes:
        - name: volrun
          hostPath:
            path: /var/run
        - name: volsys
          hostPath:
            path: /sys
        - name: voldocker
          hostPath:
            path: /var/lib/containerd/
```



Volume

- hostPath:
- Container create file

Host check file

```
praparnlueangphoonlap — Terminal MAC Pro — kubectl exec -it cadviso
~ — Terminal MAC Pro — ssh • minikube ssh
praparns-MacBook-Pro% kubectl get pods
NAME                          READY     STATUS    RESTARTS   AGE
cadvisor-1335144146-jgm3h     1/1       Running   0          15m
praparns-MacBook-Pro% kubectl exec -it cadvisor-1335144146-jgm3h sh
/ # touch /var/lib/docker/test_cAdvisor.txt
/ # ls -lh /var/lib/docker
\total 52
drwx------   28 root     root        4.0K Sep 24 15:57 containers
drwx------    3 root     root        4.0K Sep 24 04:02 image
drwxr-x---    3 root     root        4.0K Sep 24 04:02 network
drwx------  155 root     root       20.0K Sep 24 15:57 overlay
drwx------    2 root     root        4.0K Sep 24 04:02 swarm
-rw-r--r--    1 root     root           0 Sep 24 16:12 test_cAdvisor.txt
-rw-r--r--    1 root     root           0 Sep 24 16:00 testfile.txt
drwx------    2 root     root        4.0K Sep 24 15:13 tmp
drwx------    2 root     root        4.0K Sep 24 04:02 trust
drwx------    2 root     root        4.0K Sep 24 09:27 volumes
/ #
```

```
praparns-MacBook-Pro% minikube ssh
$ sudo su -
# ls /var/lib/docker
containers  image  network  overlay  swarm  test_cAdvisor.txt  testfile.txt  tmp  trust  volumes
#
```

Volume

Users > praparnlueangphoonlap > Work > Kubernetes > Training > k

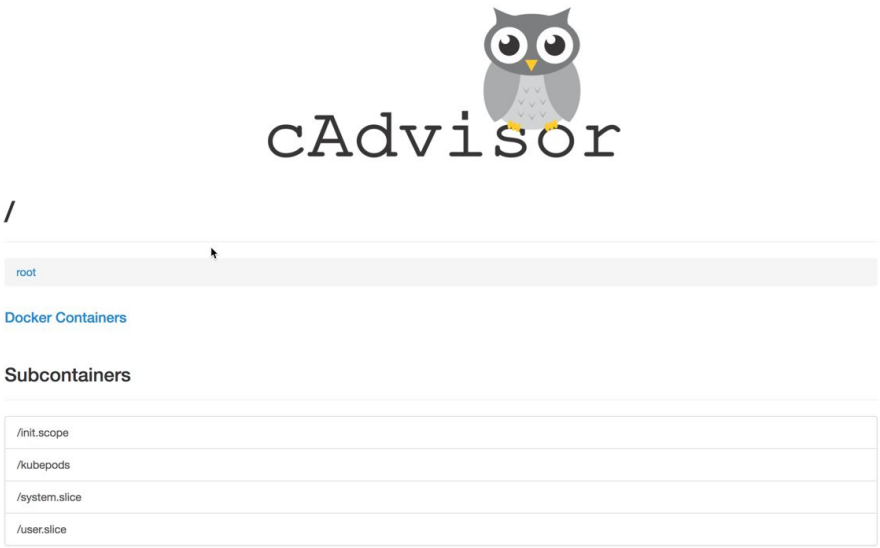
```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      ports:
        - containerPort: 5000
          protocol: TCP
      volumeMounts:
      - mountPath: /temp
        name: voldocker
  volumes:
    - name: voldocker
      hostPath:
        path: /var/lib/containerd/
```

```
praparns-MacBook-Pro% kubectl create -f webtest_pod.yml
pod "webtest" created
praparns-MacBook-Pro% kubectl get pods
NAME                          READY     STATUS    RESTARTS   AGE
cadvisor-1335144146-jgm3h     1/1       Running   0          21m
webtest                       1/1       Running   0          3s
praparns-MacBook-Pro% kubectl exec -it webtest sh
/usr/src/app # ls -lh /temp
total 56
drwx------   32 root     root        4.0K Sep 24 16:18 containers
drwx------    3 root     root        4.0K Sep 24 04:02 image
drwxr-x---    3 root     root        4.0K Sep 24 04:02 network
drwx------  163 root     root       24.0K Sep 24 16:18 overlay
drwx------    2 root     root        4.0K Sep 24 04:02 swarm
-rw-r--r--    1 root     root           0 Sep 24 16:12 test_cAdvisor.txt
-rw-r--r--    1 root     root           0 Sep 24 16:00 testfile.txt
drwx------    2 root     root        4.0K Sep 24 15:13 tmp
drwx------    2 root     root        4.0K Sep 24 04:02 trust
drwx------    2 root     root        4.0K Sep 24 09:27 volumes
/usr/src/app #
```

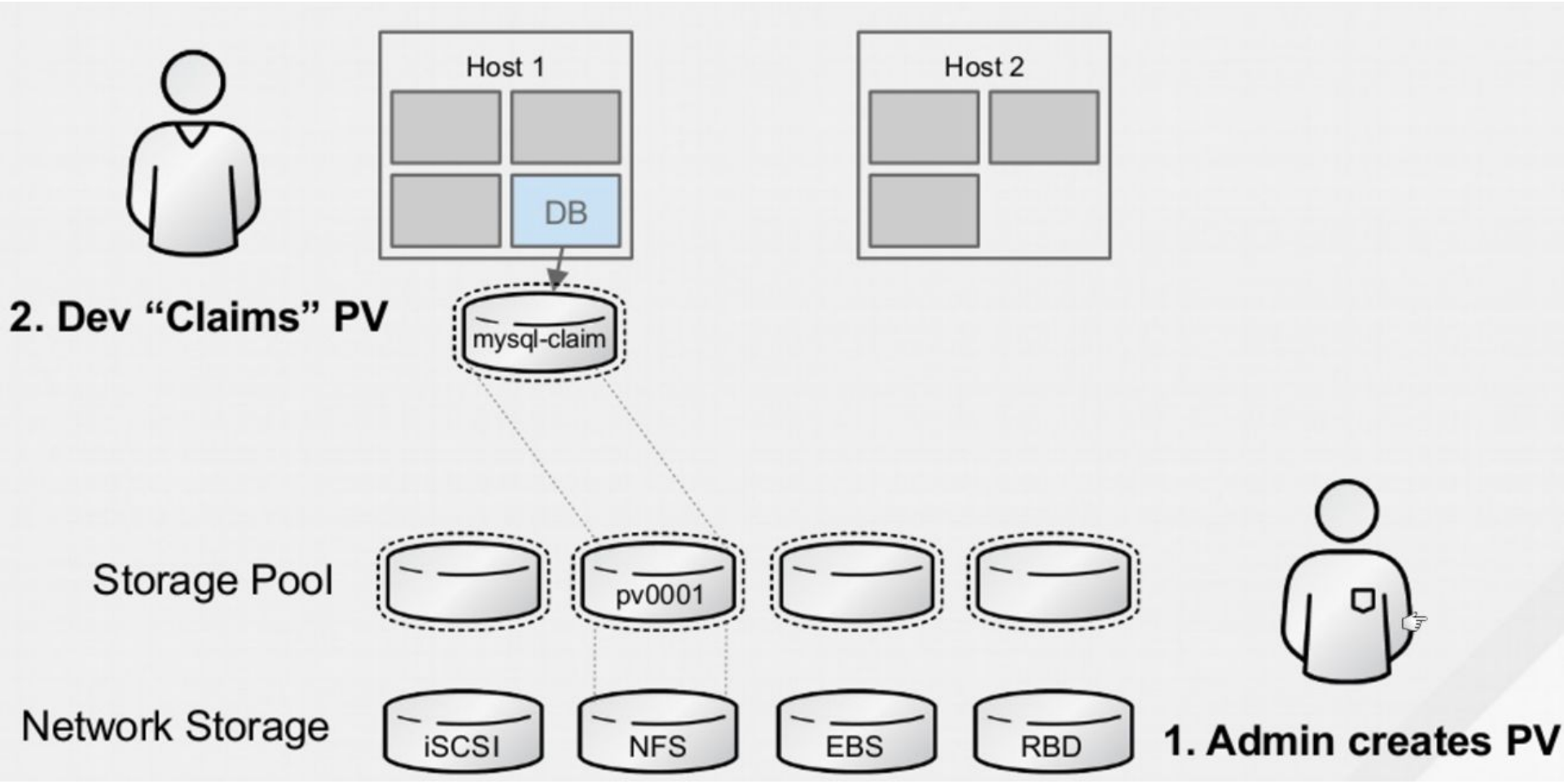
Workshop: Volume

Users > praparnlueangphoonlap > Work > Kubernetes > Train

```yaml
    labels:
      name: cadvisor
      owner: Praparn_L
      version: "1.0"
      module: Monitor
      environment: development
  spec:
    replicas: 1
    selector:
      matchLabels:
        name: cadvisor
        owner: Praparn_L
        version: "1.0"
        module: Monitor
        environment: development
    template:
      metadata:
        labels:
          name: cadvisor
          owner: Praparn_L
          version: "1.0"
          module: Monitor
          environment: development
      spec:
        containers:
          - name: cadvisor
            volumeMounts:
            - mountPath: /var/run
              name: volrun
            - mountPath: /sys
              name: volsys
            - mountPath: /var/lib/containerd/
              name: voldocker
            image: labdocker/cadvisor:latest
            ports:
            - containerPort: 8080
              protocol: TCP
        volumes:
        - name: volrun
          hostPath:
            path: /var/run
        - name: volsys
          hostPath:
            path: /sys
        - name: voldocker
          hostPath:
            path: /var/lib/containerd/
```



Volume


Host 1
DB
Host 2
2. Dev "Claims" PV
mysql-claim
Storage Pool
pv0001
Network Storage
iSCSI
NFS
EBS
RBD
1. Admin creates PV

Liveness,Readiness and Startup Probe

Kubernetes: Production Workload Orchestration

kubernetes by Google

Liveness,Readiness and Startup 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,Readiness and Startup Probe

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

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

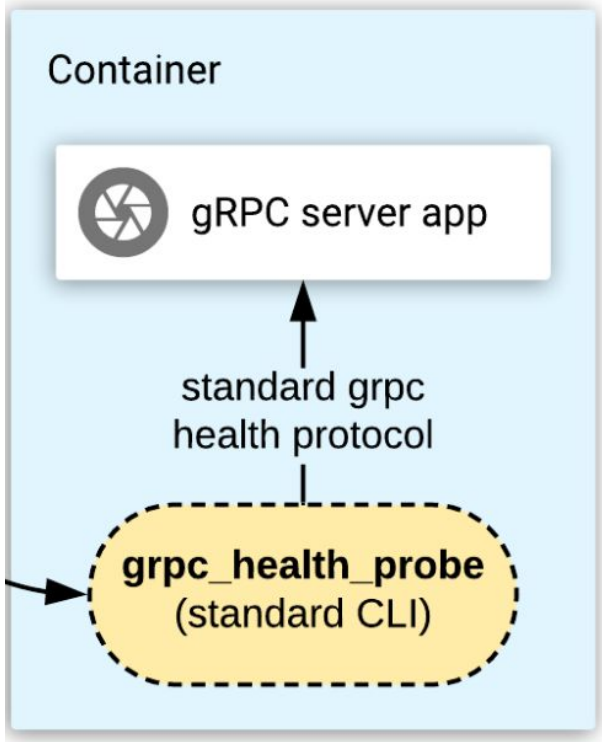
- Do we need more specific health-check?
  - Depend on container (application) fail condition.
  - If container fail condition will make process "crash" or effect to container down/unhealthy. This no need to operate more check.
  - If container fail condition may occur inside application that possible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

Liveness,Readiness and Startup Probe

- 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)
    - gRpc: Performs a remote procedure call using gRPC. The target should implement gRPC health checks. The diagnostic is considered successful if the status of the response is "SERVING".
    - 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)

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness,Readiness and Startup Probe



By implementing the standard gRPC health check protocol, you can reuse the same tool to probe all your gRPC servers.

Ref: https://kubernetes.io/blog/2018/10/01/health-checking-grpc-servers-on-kubernetes/

Liveness,Readiness and Startup Probe

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: etcd-with-grpc
spec:
  containers:
  - name: etcd
    image: registry.k8s.io/etcd:3.5.1-0
    command: [ "/usr/local/bin/etcd", "--data-dir",  "/var/lib/etcd", "--listen-client-urls", "http://0.0.0.0:2379", "--advertise-client-urls", "http://127.0.0.1:2379", "--log-level", "debug"]
    ports:
    - containerPort: 2379
    livenessProbe:
      grpc:
        port: 2379
      initialDelaySeconds: 10
```

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- Container probe process
  - Probe's result
    - Success
    - Failed
    - Unknown
  - livenessProbe, readinessProbe and startupProbe Difference
    - **livenessProbe**: check status of container is running properly or not ?
    - **readinessProbe**: check readiness for process service or not ?
    - **startupProbe**: check application in pods is start or not ? (All other probes are disabled if a startup probe is provided, until it succeeds. If the startup probe fails, the kubelet kills the container, and the container is subjected to its restart policy. If a container does not provide a startup probe, the default state is "Success")

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- Container probe process
  - Configure for probe
    - initialDelaySeconds: xxx seconds
    - periodSeconds: xxx seconds
    - timeoutSeconds: xxx seconds
    - successThreshold: 999
    - failureThreshold: 999
    - terminationGracePeriodSeconds: xxx seconds

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- Container Lifecycle state
  - PodScheduled: the Pod has been scheduled to a node.
  - PostStart: (beta feature on V1.29; enabled by default) the Pod sandbox has been successfully created and networking configured.
  - ContainersReady: all containers in the Pod are ready.
  - Initialized: all init containers have completed successfully.
  - Ready: the Pod is able to serve requests and should be added to the load balancing pools of all matching Services.

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- Container Lifecycle Hooks
  - On kubernetes we can provide some process (Exec, HTTP, Sleep (Need feature gate: PodLifecycleSleepAction=true)) 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

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

```yaml
            - name: ES_JAVA_OPTS
              value: -Xms5g -Xmx5g
            - name: bootstrap.memory_lock
              value: "false"
            - name: cluster.name
              value: pam-es-docker-cluster
            - name: xpack.security.enabled
              value: "false"
            image: "3dsinteractive/elasticsearch-std:6.6"
            lifecycle:
              postStart:
                exec:
                  command: ["/bin/sh", "-c", "chown -R 1000:1000 /usr/share/elasticsearch/data && chmod -R 777 /usr/share/elasticsearch/data"]

#           imagePullPolicy: Always
            ports:
              - name: "elastic9200"
                containerPort: 9200
              - name: "elastic9300"
                containerPort: 9300
            resources:
```

```yaml
apiVersion: app/v1
kind: Deployment
metadata:
  name: nginx
spec:
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
        lifecycle:
          preStop:
            exec:
              command: ["nginx","-s","quit"]
```

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- ExecAction:

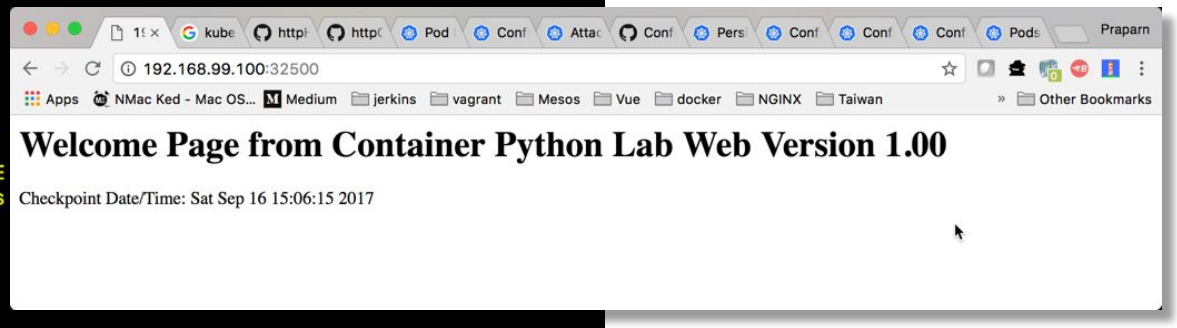
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
```

```yaml
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      ports:
      - containerPort: 5000
        protocol: TCP
      readinessProbe:
        exec:
          command:
            - cat
            - /usr/src/app/main.py
        initialDelaySeconds: 15
        periodSeconds: 5
      livenessProbe:
        exec:
          command:
            - cat
            - /usr/src/app/main.py
        initialDelaySeconds: 15
        periodSeconds: 15
      startupProbe:
        exec:
          command:
            - cat
            - /usr/src/app/main.py
        failureThreshold: 10
        successThreshold: 1
        periodSeconds: 15
```

Liveness and Readiness Probe

- ExecAction:

```
praparns-MacBook-Pro% kubectl create -f webtest_deploy_liveness_readiness_exec.yml
        kubectl create -f webtest_svc.yml
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
  Containers:
   webtest:
    Image:              labdocker/cluster:webservicelite_v1
    Port:               5000/TCP
    Liveness:           exec [cat /usr/src/app/main.py] delay=15s timeout=1s period=15s #success=1 #failure=3
    Readiness:          exec [cat /usr/src/app/main.py] delay=15s timeout=1s period=5s #success=1 #failure=3
    Environment:        <none>
```



Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- 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%
```

```
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","namespace":"default","na
me":"webtest-336910061","uid":"4992f987-9af0-11e7-bc1a-080027fb95be",...
Status:         Running
```

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- ExecAction:

```
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type            Reason                  Message
  ---------     --------        -----   ----                    -------------                   --------        ------                  -------
  7m            7m              1       default-scheduler                                       Normal          Scheduled               Successfull
y assigned webtest-336910061-26pdb to minikube
  7m            7m              1       kubelet, minikube                                       Normal          SuccessfulMountVolume   MountVolume
.SetUp succeeded for volume "default-token-pcw4h"
  2m            2m              8       kubelet, minikube       spec.containers{webtest}        Warning         Unhealthy               Readiness p
robe failed: cat: can't open '/usr/src/app/main.py': No such file or directory

  2m    2m      3       kubelet, minikube       spec.containers{webtest}        Warning Unhealthy       Liveness probe failed: cat: can't open '/us
r/src/app/main.py': No such file or directory

  7m    2m      2       kubelet, minikube       spec.containers{webtest}        Normal  Pulled  Container image "labdocker/cluster:webservicelite_v
1" already present on machine
  7m    2m      2       kubelet, minikube       spec.containers{webtest}        Normal  Created Created container
  7m    2m      2       kubelet, minikube       spec.containers{webtest}        Normal  Started Started container
  2m    2m      1       kubelet, minikube       spec.containers{webtest}        Normal  Killing Killing container with id docker://webtest:pod "web
test-336910061-26pdb_default(499cef14-9af0-11e7-bc1a-080027fb95be)" container "webtest" is unhealthy, it will be killed and re-created.
praparns-MacBook-Pro%
```

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

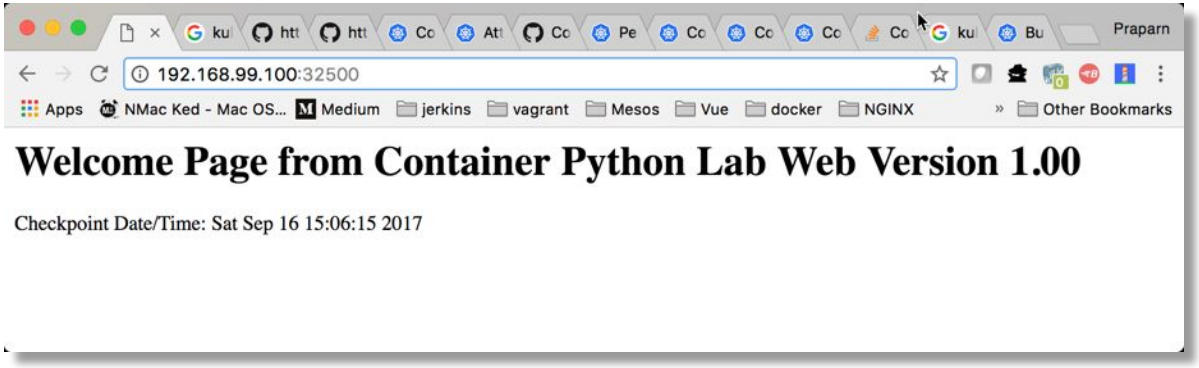
- TCPSocketAction:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
```

```yaml
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - name: webservice
            containerPort: 5000
            protocol: TCP
          readinessProbe:
            tcpSocket:
              port: 5000
            initialDelaySeconds: 15
            periodSeconds: 5
          livenessProbe:
            tcpSocket:
              port: 5000
            initialDelaySeconds: 15
            periodSeconds: 15
          startupProbe:
            tcpSocket:
              port: 5000
            successThreshold: 1
            failureThreshold: 10
            periodSeconds: 30
```

Liveness and Readiness Probe

- TCPSocketAction:




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

```
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type          Reason                  Message
  ---------     --------        -----   ----                    -------------                   --------      ------                  -------
  2m            2m              1       default-scheduler                                       Normal        Scheduled               Successfull
y assigned webtest-3579274848-1w2mn to minikube
  2m            2m              1       kubelet, minikube                                       Normal        SuccessfulMountVolume   MountVolume
.SetUp succeeded for volume "default-token-pcw4h"
  2m            2m              1       kubelet, minikube       spec.containers{webtest}        Normal        Pulled                  Container i
mage "labdocker/cluster:webservicelite_v1" already present on machine
  2m            2m              1       kubelet, minikube       spec.containers{webtest}        Normal        Created                 Created con
tainer
  2m            2m              1       kubelet, minikube       spec.containers{webtest}        Normal        Started                 Started con
tainer
```

Ref: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/

Liveness and Readiness Probe

- TCPSocketAction:

instruction.txt | webtest_deploy_liveness_readiness_port.yml

```yaml
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      ports:
      - name: webservice
        containerPort: 5000
        protocol: TCP
      readinessProbe:
        tcpSocket:
          port: 3000
        initialDelaySeconds: 15
        periodSeconds: 5
      livenessProbe:
        tcpSocket:
          port: 3000
        initialDelaySeconds: 15
        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"...
```

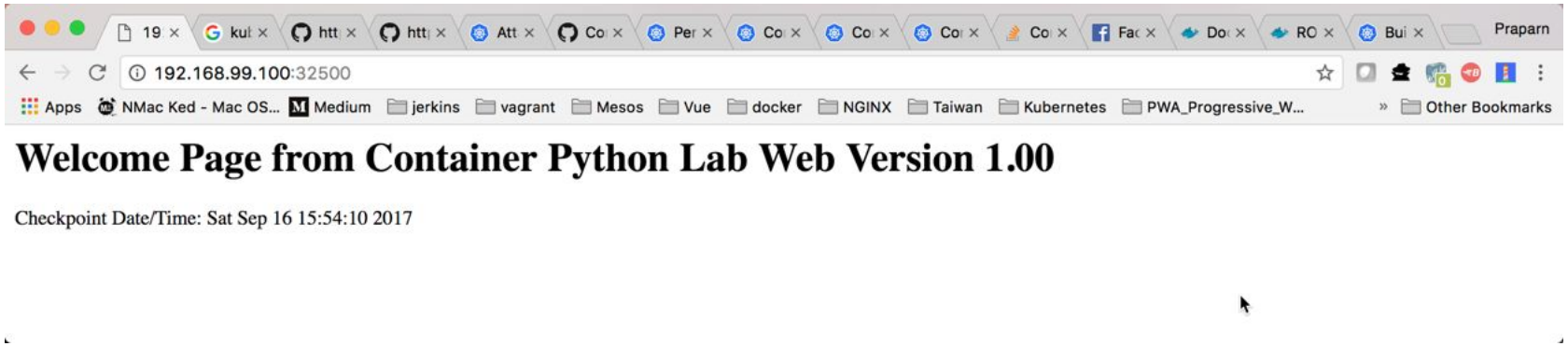
Liveness and Readiness Probe

- TCPSocketAction:

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

Liveness and Readiness Probe

- HTTPGetAction:



Liveness and Readiness Probe

- HTTPGetAction:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
```

```yaml
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      ports:
      - name: webservice
        containerPort: 5000
        protocol: TCP
      readinessProbe:
        httpGet:
          # Optional "host: my-host" for set specific ip of Pods
          # Optional "scheme: HTTPS"
          #schema: HTTPS
          path: /
          port: webservice
          httpHeaders:
          - name: Server
            value: Werkzeug/0.12.2 Python/2.7.13
        initialDelaySeconds: 15
        periodSeconds: 5
        timeoutSeconds: 10
        successThreshold: 1
        failureThreshold: 3
      livenessProbe:
        httpGet:
          # Optional "host: my-host" for set specific ip of Pods
          # Optional "scheme: HTTPS"
          #schema: HTTPS
          path: /
          port: webservice
          httpHeaders:
          - name: Server
            value: Werkzeug/0.12.2 Python/2.7.13
        initialDelaySeconds: 15
        periodSeconds: 30
        timeoutSeconds: 30
        successThreshold: 1
        failureThreshold: 10
```

Liveness and Readiness Probe

- HTTPGetAction:

```yaml
startupProbe:
  httpGet:
    # Optional "host: my-host" for set specific ip of Pods
    # Optional "scheme: HTTPS"
    #schema: HTTPS
    path: /
    port: webservice
    httpHeaders:
    - name: Server
      value: Werkzeug/0.12.2 Python/2.7.13
  periodSeconds: 30
  timeoutSeconds: 30
  successThreshold: 1
  failureThreshold: 10
```

Liveness and Readiness Probe

- HTTPGetAction:

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

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

Liveness and Readiness Probe

- HTTPGetAction:

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

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

Liveness and Readiness Probe

- HTTPGetAction:

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

Liveness and Readiness Probe

- ReadinessGate:

Users > praparnlueangphoonlap > Desktop > ! 1.yaml

```yaml
kind: Pod
...
spec:
  readinessGates:
    - conditionType: "www.example.com/feature-1"
status:
  conditions:
    - type: Ready                              # a built in PodCondition
      status: "False"
      lastProbeTime: null
      lastTransitionTime: 2018-01-01T00:00:00Z
    - type: "www.example.com/feature-1"        # an extra PodCondition
      status: "False"
      lastProbeTime: null
      lastTransitionTime: 2018-01-01T00:00:00Z
  containerStatuses:
    - containerID: docker://abcd...
      ready: true
...
```

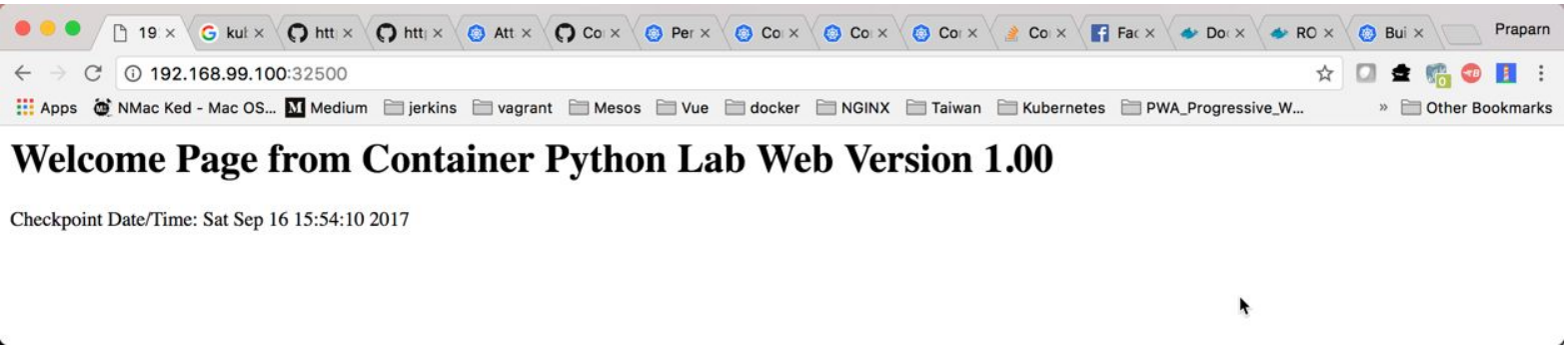
Liveness and Readiness Probe

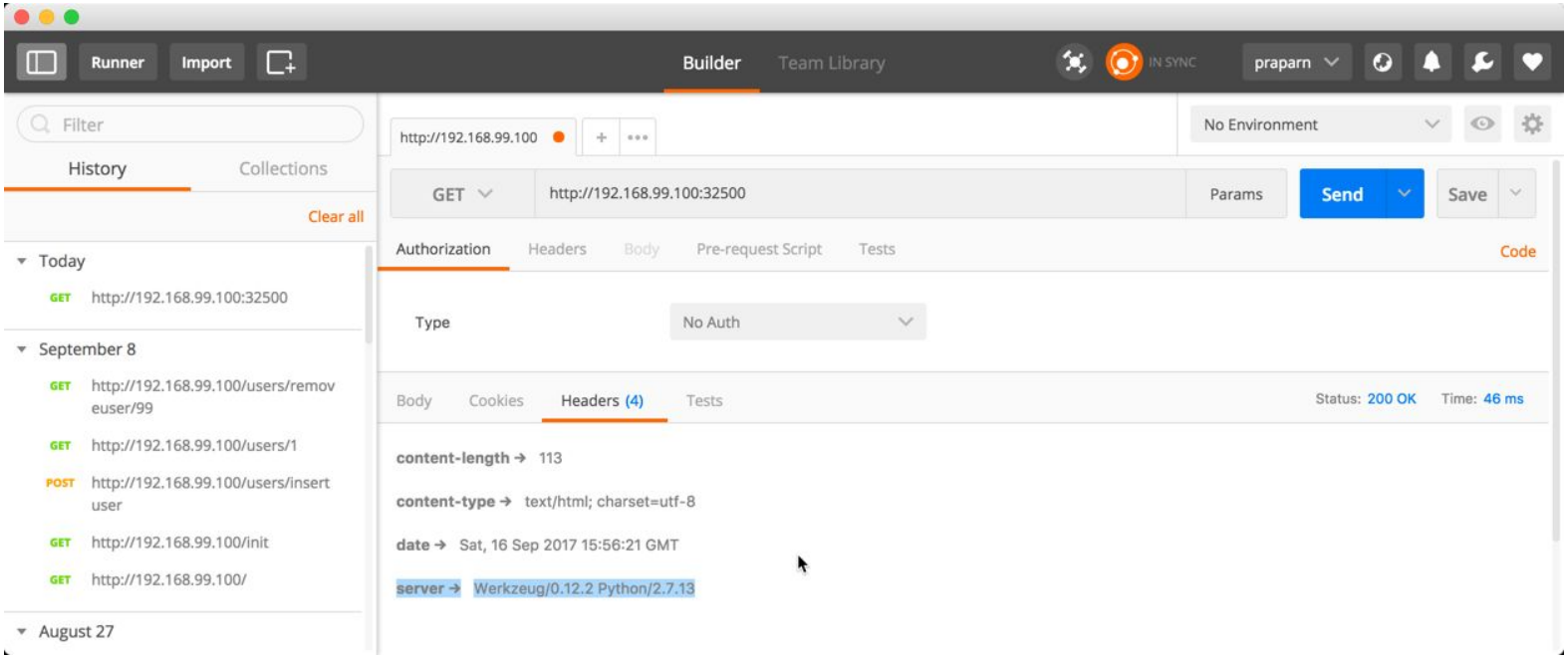
- gRPCAction:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: registry.k8s.io/etcd:3.5.1-0
          command: [ "/usr/local/bin/etcd", "--data-dir",  "/var/lib/etcd",
          "--listen-client-urls", "http://0.0.0.0:2379",
          "--advertise-client-urls", "http://127.0.0.1:2379",
          "--log-level", "debug"]
```

```yaml
          ports:
          - containerPort: 2379
            protocol: TCP
          readinessProbe:
            grpc:
              port: 2379
            initialDelaySeconds: 15
            periodSeconds: 5
          livenessProbe:
            grpc:
              port: 2379
            initialDelaySeconds: 15
            periodSeconds: 15
          startupProbe:
            grpc:
              port: 2379
            successThreshold: 1
            failureThreshold: 10
            periodSeconds: 30
```

Workshop: Healthcheck Probe



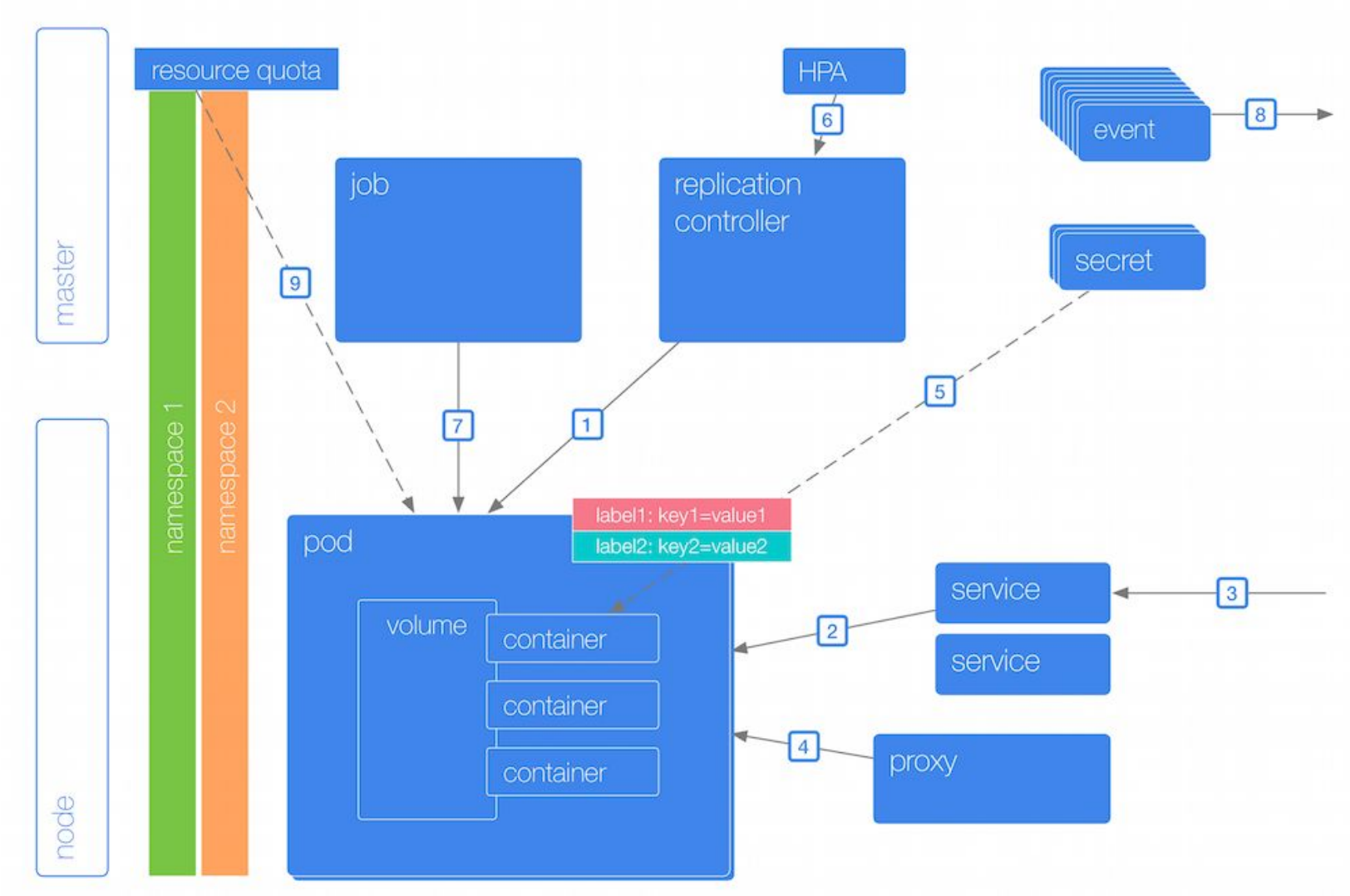


Resource Management and HPA

Kubernetes: Production Workload Orchestration

kubernetes by Google

Resource Management and HPA



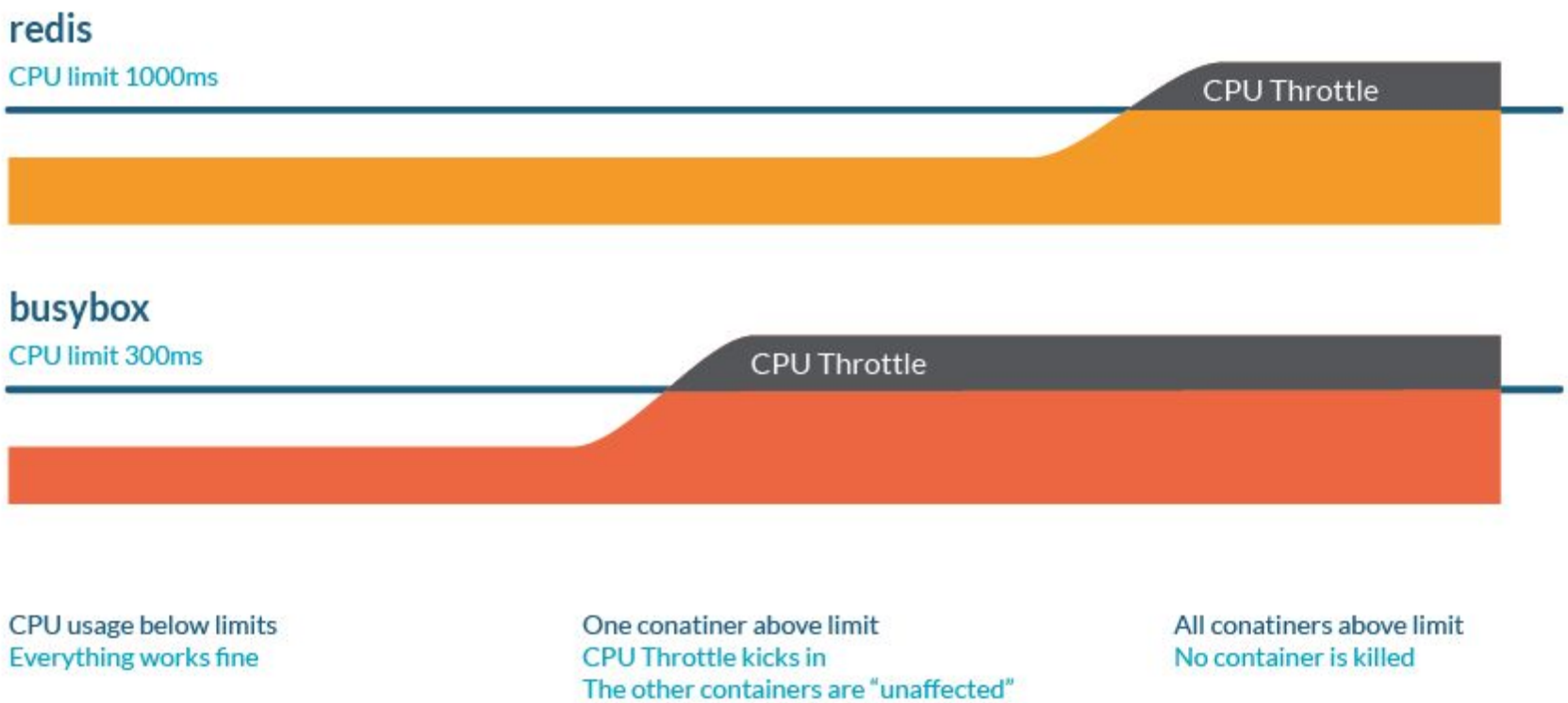
Ref: http://k8s.info/cs.html

Resource Management and HPA

- Kubernetes can manage resource in 2 ways
  - Container level configuration
    - Scope on Container unit independence
    - etc
  - Namespace level configuration
    - Scope on Pods and Container level
    - Create global limit and assign to namespace
    - Apply namespace to any Pods (Template Resource Limit)
- What happen when Container resource exceed ?
  - CPU
    - Container will not get CPU as it need and the free-slot of CPU was distributed to all container's request with ratio
  - Memory
    - Container was killed and restart it again
  - Ephemeral-storage (Emptydir, configmap, CSI ephemeral volumes, generic ephemeral)
    - Eviction (kill to new pods)(revenge !!!)
  - Storage
    - Cannot use more space on storage (depend on application)

Resource Management and HPA

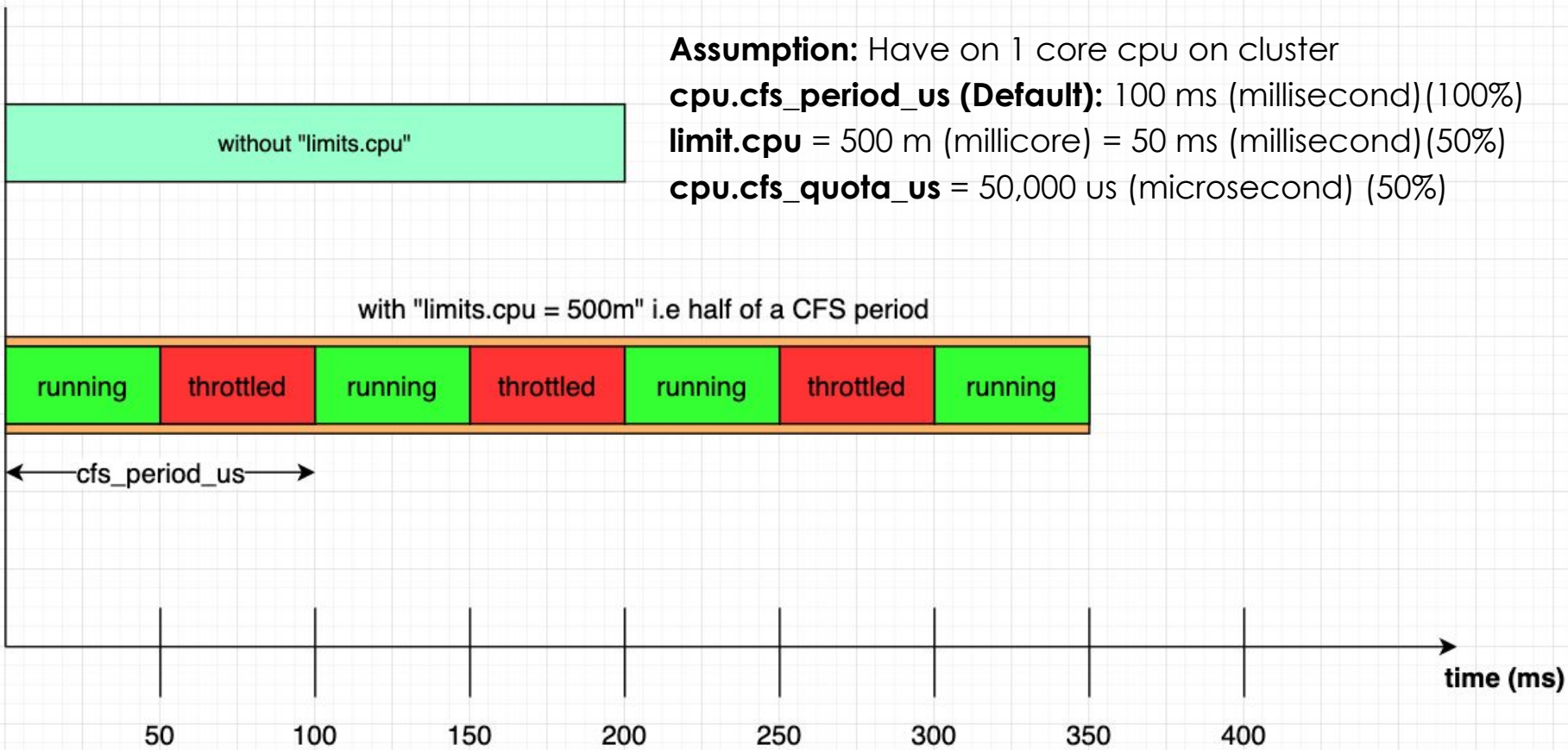
- CPU Limit and Throttling



Ref:https://sysdig.com/blog/troubleshoot-kubernetes-oom/

Resource Management and HPA

- CPU Limit and Throttling



Assumption: Have on 1 core cpu on cluster
cpu.cfs_period_us (Default): 100 ms (millisecond)(100%)
limit.cpu = 500 m (millicore) = 50 ms (millisecond)(50%)
cpu.cfs_quota_us = 50,000 us (microsecond) (50%)

Ref:https://jaanhio.me/blog/kubernetes-cpu-requests-limits/

Resource Management and HPA

- CPU Limit and Throttling

Container	CPU Usage	CPU Requests	CPU Requests %	CPU Limits	CPU Limits %
consul	0.04	0.05	87.47%	0.10	43.74%

Ref:https://jaanhio.me/blog/kubernetes-cpu-requests-limits/

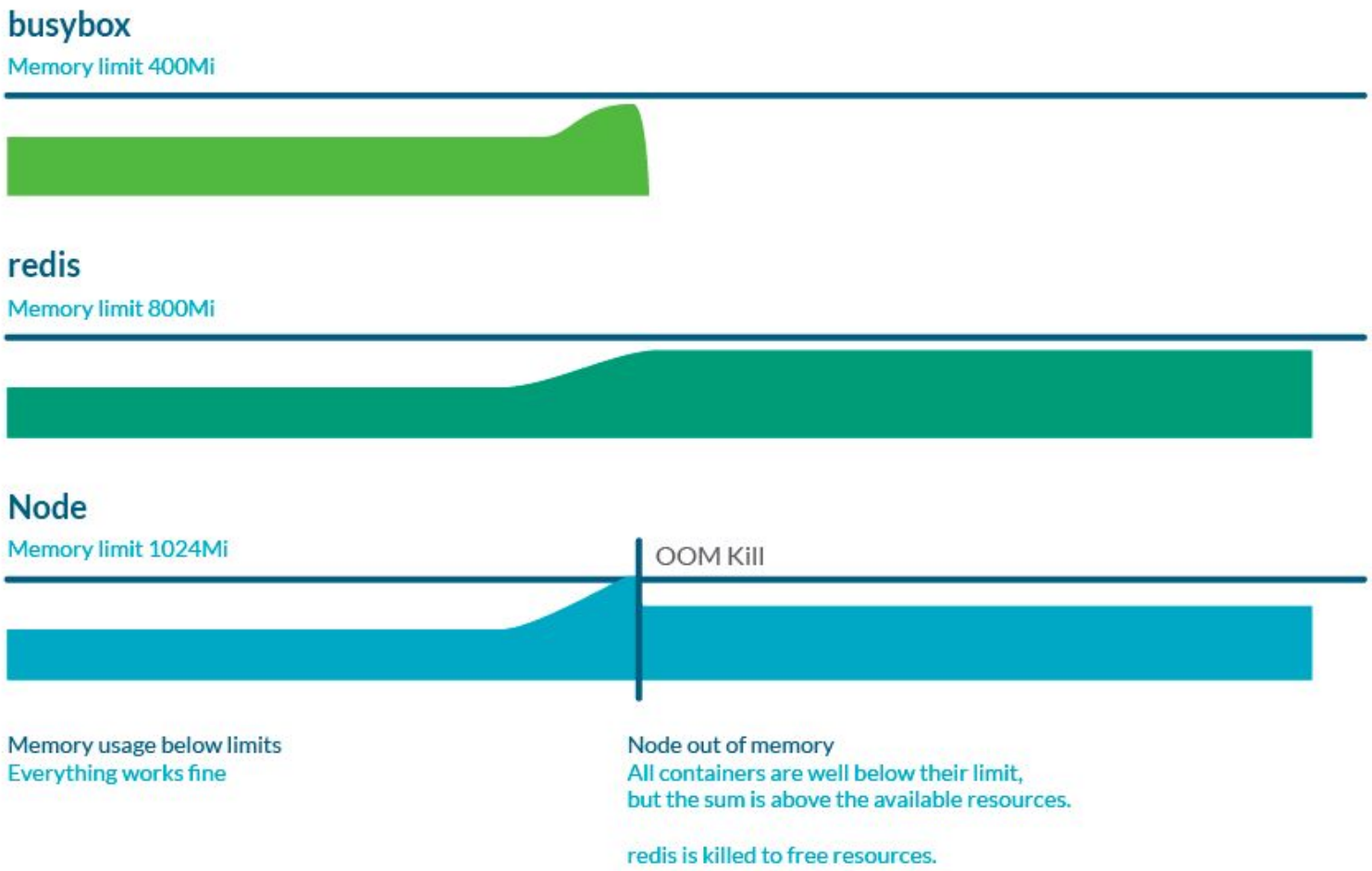
Resource Management and HPA

- CPU Limit and Throttling

Ref:https://www.ibm.com/blog/kubernetes-cpu-throttling-the-silent-killer-of-response-time/

Resource Management and HPA

- Memory Limit and OOM Kill



Ref:https://sysdig.com/blog/troubleshoot-kubernetes-oom/

Resource Management and HPA

- Container level configuration
  - CPU
    - Request:
      - XXXm(Unit: millicores) (Ratio: 1024)
      - 0.1= 100m
      - 1 CPU =
        - 1 vCPU (AWS)
        - 1 GCP (Google Cloud)
        - 1 Azure v Core
        - 1 Hyper thread (Bare Meterial)
      - Consider as "--cpu-share"
    - Limit:
      - XXXm(Unit: millicores) (Ratio:100000/1000: ~1000)
      - Consider as "--cpu-quota"
  - Memory
    - Request:
      - XXX (Unit:Ki, Mi, Gi, Pi, Ei)
    - Limit
      - XXX (Unit:Ki, Mi, Gi, Pi, Ei)

Ref: https://github.com/kubernetes/community/blob/master/contributors/design-proposals/resource-qos.md

Resource Management and HPA

- Container level configuration
  - Pods “webtest”

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      resources:
        requests:
          memory: "16Mi"
          cpu: "100m"
          ephemeral-storage: 100Mi
        limits:
          memory: "32Mi"
          cpu: "1"
          ephemeral-storage: 100Mi
      ports:
        - containerPort: 5000
          protocol: TCP
```

System status

```
...ckage/kubernetes — Terminal MAC Pro — zsh    docker@iZj6chay173lo9ivaj1drnZ: ~ — -bash    Terminal MAC Pro — ssh ◂ minikube ssh
top - 07:39:14 up  4:10,  1 user,  load average: 0.06, 0.29, 0.34
Tasks: 233 total,   1 running, 232 sleeping,   0 stopped,   0 zombie
%Cpu0  :   0.0/0.0     0[                                                    ]
%Cpu1  :   0.0/0.0     0[                                                    ]
%Cpu2  :   0.7/0.7     1[                                                    ]
%Cpu3  :   0.0/0.0     0[                                                    ]
%Cpu4  :   0.0/0.0     0[                                                    ]
%Cpu5  :   0.0/0.0     0[                                                    ]
%Cpu6  :   0.7/0.7     1[                                                    ]
%Cpu7  :   0.0/1.3     1[|                                                   ]
%Cpu8  :   0.0/0.0     0[                                                    ]
%Cpu9  :   0.7/0.7     1[                                                    ]
GiB Mem : 30.2/1.953    [                                                    ]
GiB Swap:  0.0/0.977    [                                                    ]

  PID USER      PR  NI    VIRT    RES  %CPU %MEM     TIME+ S COMMAND
    1 root      20   0   37.8m   5.9m   0.0  0.3   0:14.97 S systemd
 1440 root      20   0   65.9m  20.9m   0.0  1.0   0:06.73 S  `- systemd-journal
 2311 root      20   0   25.6m   3.2m   0.0  0.2   0:01.81 S  `- systemd-udevd
 2608 root      20   0    8.9m   0.4m   0.0  0.0   0:00.46 S  `- getty
 2617 root      20   0   24.2m   2.0m   0.0  0.1   0:00.02 S  `- rpcbind
 2638 root      20   0   25.5m   2.8m   0.0  0.1   0:00.17 S  `- systemd-logind
```

Resource Management and HPA

- Container level configuration
  - Allocate status on node

```
  Namespace                   Name                                     CPU Requests  CPU Limits  Memory Requests  Memory Limits
  ---------                   ----                                     ------------  ----------  ---------------  -------------
  default                     cadvisor-76c9899d7f-pbz29                0 (0%)        0 (0%)      0 (0%)           0 (0%)
  default                     webtest                                  100m (1%)     1 (10%)     16Mi (0%)        32Mi (1%)
  kube-system                 default-http-backend-xq22v               10m (0%)      10m (0%)    20Mi (1%)        20Mi (1%)
  kube-system                 heapster-4nfdm                           0 (0%)        0 (0%)      0 (0%)           0 (0%)
  kube-system                 influxdb-grafana-kzmsf                   0 (0%)        0 (0%)      0 (0%)           0 (0%)
  kube-system                 kube-addon-manager-minikube              5m (0%)       0 (0%)      50Mi (2%)        0 (0%)
  kube-system                 kube-dns-54cccfbdf8-tqh5l                260m (2%)     0 (0%)      110Mi (5%)       170Mi (8%)
  kube-system                 kubernetes-dashboard-77d8b98585-z4lxz    0 (0%)        0 (0%)      0 (0%)           0 (0%)
  kube-system                 nginx-ingress-controller-hjwt6           0 (0%)        0 (0%)      0 (0%)           0 (0%)
  kube-system                 storage-provisioner                      0 (0%)        0 (0%)      0 (0%)           0 (0%)
Allocated resources:
```

Resource Management and HPA

- Container level configuration
  - Create Load on CPU (T1)

```
praparns-MacBook-Pro% kubectl create -f webtest_pod.yml
pod "webtest" created
praparns-MacBook-Pro% kubectl get pods
NAME      READY     STATUS    RESTARTS   AGE
webtest   1/1       Running   0          13m
praparns-MacBook-Pro% kubectl exec webtest -c webtest md5sum /dev/urandom
^C
praparns-MacBook-Pro%
```

  - System status

```
top - 07:45:00 up  4:16,  1 user,  load average: 0.69, 0.35, 0.33
Tasks: 234 total,   2 running, 232 sleeping,   0 stopped,   0 zombie
%Cpu0  :   0.0/0.0     0[                                                    ]
%Cpu1  :   0.0/1.3     1[|                                                   ]
%Cpu2  :   0.0/0.0     0[                                                    ]
%Cpu3  :  26.2/73.8  100[||||||||||||||||||||||||||||||||||||||||||||||||||||]
%Cpu4  :   0.7/0.0     1[                                                    ]
%Cpu5  :   0.7/0.0     1[                                                    ]
%Cpu6  :   0.7/0.0     1[                                                    ]
%Cpu7  :   0.7/0.7     1[                                                    ]
%Cpu8  :   0.7/0.0     1[                                                    ]
%Cpu9  :   0.7/0.7     1[                                                    ]
GiB Mem : 31.2/1.953    [                                                    ]
GiB Swap:  0.0/0.977    [                                                    ]
```

Resource Management and HPA

- Container level configuration
  - Create Load on CPU (T2)

```
praparns-MacBook-Pro% kubectl create -f webtest_pod.yml
pod "webtest" created
praparns-MacBook-Pro% kubectl get pods
NAME       READY     STATUS    RESTARTS   AGE
webtest    1/1       Running   0          13m
praparns-MacBook-Pro% kubectl exec webtest -c webtest md5sum /dev/urandom
^C
praparns-MacBook-Pro% kubectl exec webtest -c webtest md5sum /dev/urandom
^C
praparns-MacBook-Pro%
```

  - System status

```
top - 07:46:38 up  4:17,  1 user,  load average: 0.61, 0.47, 0.38
Tasks: 236 total,   3 running, 233 sleeping,   0 stopped,   0 zombie
%Cpu0  :   0.0/0.0     0[                                                  ]
%Cpu1  :   3.4/0.7     4[||                                                ]
%Cpu2  :   2.0/0.0     2[|                                                 ]
%Cpu3  :   6.8/43.2   50[||||||||||||||||||||||||||||||||||                ]
%Cpu4  :   2.0/0.7     3[|                                                 ]
%Cpu5  :   2.0/1.3     3[||                                                ]
%Cpu6  :   0.0/0.0     0[                                                  ]
%Cpu7  :   6.0/43.6   50[||||||||||||||||||||||||||||||||||                ]
%Cpu8  :   3.5/0.7     4[|||                                               ]
%Cpu9  :   2.1/1.4     3[||                                                ]
GiB Mem : 31.2/1.953    [                                                  ]
GiB Swap:  0.0/0.977    [                                                  ]
```

Workshop: Resource MNG/HPA

- Part 1: Container level configuration

```
top - 07:45:00 up  4:16,  1 user,  load average: 0.69, 0.35, 0.33
Tasks: 234 total,   2 running, 232 sleeping,   0 stopped,   0 zombie
%Cpu0  :   0.0/0.0     0[                                                  ]
%Cpu1  :   0.0/1.3     1[|                                                 ]
%Cpu2  :   0.0/0.0     0[                                                  ]
%Cpu3  :  26.2/73.8  100[||||||||||||||||||||||||||||||||||||||||||||||||||]
%Cpu4  :   0.7/0.0     1[                                                  ]
%Cpu5  :   0.7/0.0     1[                                                  ]
%Cpu6  :   0.7/0.0     1[                                                  ]
%Cpu7  :   0.7/0.7     1[                                                  ]
%Cpu8  :   0.7/0.0     1[                                                  ]
%Cpu9  :   0.7/0.7     1[                                                  ]
GiB Mem : 31.2/1.953    [                                                  ]
GiB Swap:  0.0/0.977    [                                                  ]
```

Resource Management and HPA

- Namespace level configuration
  - Name space provide collection of resource of cluster system that easy to defined and apply to object in cluster system
  - Name space can apply with
    - Pods
    - Services
    - Replication Controller
    - Deployment and ReplicaSets
    - Quota
    - LimitRange
    - Etc
  - For resource management resource in name space level. We will use Quota and LimitRange
    - Quota: Summary of resource control on name space
    - LimitRange: Resource admission control/Default resource define

Resource Management and HPA

- Quota
  - Compute resource
    - CPU
      - Limits
      - Request
    - Memory
      - Limits
      - Request
    - Ephemeral-storage
      - Limits
      - Request
    - Storage
      - Limits
      - Request
  - Storage-class
    - <storage-class-name>.storageclass.storage.k8s.io/requests.storage
    - <storage-class-name>.storageclass.storage.k8s.io/persistentvolumeclaims

Ref: https://kubernetes.io/docs/concepts/policy/resource-quotas

Resource Management and HPA

- Quota
  - Counting Object
    - pods
    - services
    - services.loadbalancers
    - services.nodeports
    - replicationcontrollers
    - resourcequotas
    - configmap
    - secrets

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: webtest-quota
  labels:
    name: webtest_quota
    owner: Praparn_L
    version: "1.0"
    module: Quota
    environment: development
spec:
  hard:
    pods: "4"
    requests.cpu: "1"
    requests.memory: 1Gi
    limits.cpu: "4"
    limits.memory: 4Gi
    persistentvolumeclaims: "5"
    requests.storage: "5Gi"
    requests.ephemeral-storage: "10Gi"
    limits.ephemeral-storage: "20Gi"
```

Ref: https://kubernetes.io/docs/concepts/policy/resource-quotas

Resource Management and HPA

- Quota per priorityClass

```yaml
apiVersion: v1
kind: List
items:
- apiVersion: v1
  kind: ResourceQuota
  metadata:
    name: pods-high
  spec:
    hard:
      cpu: "1000"
      memory: 200Gi
      pods: "10"
    scopeSelector:
      matchExpressions:
      - operator : In
        scopeName: PriorityClass
        values: ["high"]
- apiVersion: v1
  kind: ResourceQuota
  metadata:
    name: pods-medium
  spec:
    hard:
      cpu: "10"
      memory: 20Gi
      pods: "10"
    scopeSelector:
      matchExpressions:
      - operator : In
        scopeName: PriorityClass
        values: ["medium"]
```

```yaml
- apiVersion: v1
  kind: ResourceQuota
  metadata:
    name: pods-low
  spec:
    hard:
      cpu: "5"
      memory: 10Gi
      pods: "10"
    scopeSelector:
      matchExpressions:
      - operator : In
        scopeName: PriorityClass
        values: ["low"]
```

```
Name:        pods-high
Namespace:   default
Resource     Used  Hard
--------     ----  ----
cpu          0     1k
memory       0     200Gi
pods         0     10


Name:        pods-low
Namespace:   default
Resource     Used  Hard
--------     ----  ----
cpu          0     5
memory       0     10Gi
pods         0     10


Name:        pods-medium
Namespace:   default
Resource     Used  Hard
--------     ----  ----
cpu          0     10
memory       0     20Gi
pods         0     10
```

Ref: https://kubernetes.io/docs/concepts/policy/resource-quotas

Resource Management and HPA

- Quota per priorityClass

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: high-priority
spec:
  containers:
  - name: high-priority
    image: ubuntu
    command: ["/bin/sh"]
    args: ["-c", "while true; do echo hello; sleep 10;done"]
    resources:
      requests:
        memory: "10Gi"
        cpu: "500m"
      limits:
        memory: "10Gi"
        cpu: "500m"
  priorityClassName: high
```

Ref: https://kubernetes.io/docs/concepts/policy/resource-quotas

Resource Management and HPA

- LimitRange
  - Type: (Pods/Container)
    - Max
      - CPU:
      - Memory:
      - Ephemeral-storage:
      - PersistentVolumeClaim:
    - Min
      - CPU
      - Memory
      - Ephemeral-storage:
      - PersistentVolumeClaim:
  - Default (Limit Default for Container)
    - CPU
    - Memory
    - Ephemeral-storage
  - DefaultRequest (Default for Container)
    - CPU
    - Memory
    - Ephemeral-storage

Ref: https://kubernetes.io/docs/tasks/administer-cluster/apply-resource-quota-limit/

Resource Management and HPA

- LimitRange

```yaml
apiVersion: v1
kind: LimitRange
metadata:
  name: webtest-limit
  labels:
    name: webtest_limit
    owner: Praparn_L
    version: "1.0"
    module: LimitRange
    environment: development
spec:
  limits:
  - type: Pod
    max:
      cpu: "1"
      memory: 1Gi
    min:
      cpu: 200m
      memory: 6Mi
```

```yaml
- type: Container
  default:
    cpu: 300m
    memory: 200Mi
    ephemeral-storage: 50Mi
  defaultRequest:
    cpu: 200m
    memory: 100Mi
    ephemeral-storage: 10Mi
  max:
    cpu: "1"
    memory: 1Gi
    ephemeral-storage: 500Mi
  min:
    cpu: 100m
    memory: 3Mi
    ephemeral-storage: 10Mi

- type: PersistentVolumeClaim
  max:
    storage: 50Gi
  min:
    storage: 1Gi
```

Ref: https://kubernetes.io/docs/tasks/administer-cluster/apply-resource-quota-limit/

Resource Management and HPA

- Create Namespace, assign quota with name space

kubectl create namespace <name>

kubectl create –f <Quota File> -- namespace <name>

```
praparns-MacBook-Pro% kubectl create namespace webtest-namespace
namespace "webtest-namespace" created
praparns-MacBook-Pro% kubectl create -f webtest_quota.yml --namespace=webtest-namespace
resourcequota "webtest-quota" created
praparns-MacBook-Pro% kubectl describe namespace webtest-namespace
Name:           webtest-namespace
Labels:         <none>
Annotations:    <none>
Status:         Active

Resource Quotas
 Name:                   webtest-quota
 Resource                Used    Hard
 --------                ---     ---
 limits.cpu              0       4
 limits.memory           0       4Gi
 pods                    0       4
 requests.cpu            0       1
 requests.memory         0       1Gi

No resource limits.
praparns-MacBook-Pro%
```

Ref: https://kubernetes.io/docs/concepts/policy/resource-quotas

Resource Management and HPA

- Deployment

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

  type: NodePort
  ports:
   - port: 5000
     name: http
     targetPort: 5000
     protocol: TCP
     nodePort: 32500
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
```

```yaml
    module: WebServer
    environment: development
spec:
  replicas: 1
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - containerPort: 5000
            protocol: TCP
```

Resource Management and HPA

- Create Deployment and check result

kubectl create –f <Deployment File> --namespace <name>

```
praparns-MacBook-Pro% kubectl create -f webtest_deploy.yml --namespace=webtest-namespace
service "webtest" created
deployment "webtest" created
praparns-MacBook-Pro% kubectl get deployment/webtest --namespace=webtest-namespace
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   1         0         0            0           5m
praparns-MacBook-Pro% kubectl get rs --namespace=webtest-namespace
NAME                 DESIRED   CURRENT   READY     AGE
webtest-4261491039   1         0         0         5m
praparns-MacBook-Pro% kubectl get svc/webtest --namespace=webtest-namespace
NAME      CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
webtest   10.0.0.48    <nodes>       5000:32500/TCP   5m
praparns-MacBook-Pro% kubectl get pods --namespace=webtest-namespace
No resources found.
praparns-MacBook-Pro%
```

Resource Management and HPA

- Why it fail ?

kubectl describe rs --namespace <name>

```
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath   Type            Reason          Message
  ---------     --------        -----   ----                    -------------   --------        ------          -------
  6m            5m              15      replicaset-controller                   Warning         FailedCreate    Error creating: pods "webtest-4261491039-" is
 forbidden: failed quota: webtest-quota: must specify limits.cpu,limits.memory,requests.cpu,requests.memory
```

```yaml
spec:
  replicas: 1
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - containerPort: 5000
            protocol: TCP
```

Resource Management and HPA

- Create LimitRange and check result

kubectl create –f <LimitRange File> --namespace <name>

```
praparns-MacBook-Pro% kubectl create -f webtest_limit.yml --namespace=webtest-namespace
limitrange "webtest-limit" created
praparns-MacBook-Pro% kubectl describe namespace/webtest-namespace
Name:           webtest-namespace
Labels:         <none>
Annotations:    <none>
Status:         Active

Resource Quotas
 Name:                   webtest-quota
 Resource                Used    Hard
 --------                ---     ---
 limits.cpu              0       4
 limits.memory           0       4Gi
 pods                    0       4
 requests.cpu            0       1
 requests.memory         0       1Gi

Resource Limits
 Type          Resource        Min     Max     Default Request Default Limit   Max Limit/Request Ratio
 ----          --------        ---     ---     --------------- -------------   -----------------------
 Pod           cpu             200m    1       -               -               -
 Pod           memory          6Mi     1Gi     -               -               -
 Container     cpu             100m    1       200m            300m            -
 Container     memory          3Mi     1Gi     100Mi           200Mi           -
praparns-MacBook-Pro%
```

Resource Management and HPA

- Recreate Deployment and check result

```
praparns-MacBook-Pro% kubectl delete -f webtest_deploy.yml --namespace=webtest-namespace
service "webtest" deleted
deployment "webtest" deleted
praparns-MacBook-Pro% kubectl create -f webtest_deploy.yml --namespace=webtest-namespace
service "webtest" created
deployment "webtest" created
praparns-MacBook-Pro% kubectl get deployment/webtest --namespace=webtest-namespace
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   1         1         1            1           5m
praparns-MacBook-Pro% kubectl get rs --namespace=webtest-namespace
NAME                 DESIRED   CURRENT   READY     AGE
webtest-4261491039   1         1         1         5m
praparns-MacBook-Pro% kubectl get svc/webtest --namespace=webtest-namespace
NAME      CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
webtest   10.0.0.125   <nodes>       5000:32500/TCP   5m
praparns-MacBook-Pro% kubectl get pods --namespace=webtest-namespace
NAME                       READY     STATUS    RESTARTS   AGE
webtest-4261491039-pft5m   1/1       Running   0          5m
praparns-MacBook-Pro% curl http://192.168.99.100:32500
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Sat Jul  8 13:44:12 2017
praparns-MacBook-Pro%
```

Resource Management and HPA

- Check describe of Pods

kubectl describe pods --namespace <name>

```
Status:         Running
IP:             172.17.0.2
Controllers:    ReplicaSet/webtest-4261491039
Containers:
  webtest:
    Container ID:       docker://0acf26d00b7b9f28b200fdf6
    Image:              labdocker/cluster:webservicelite_
    Image ID:           docker-pullable://labdocker/clust
    Port:               5000/TCP
    State:              Running
      Started:          Sat, 08 Jul 2017 20:43:27 +0700
    Ready:              True
    Restart Count:      0
    Limits:
      cpu:      300m
      memory:   200Mi
    Requests:
      cpu:              200m
      memory:           100Mi
    Environment:        <none>
```

Resource Management and HPA

- Test burn cpu utilize and monitor

```
praparns-MacBook-Pro% kubectl exec -it webtest-4261491039-pft5m -c webtest md5sum /dev/urandom --namespace=webtest-namespace
```

```
top - 13:57:03 up  7:33,  1 user,  load average: 0.22, 0.35, 0.20
Tasks: 228 total,   2 running, 226 sleeping,   0 stopped,   0 zombie
%Cpu0  :   0.0/0.0     0[                                                  ]
%Cpu1  :   0.7/0.7     1[||                                                ]
%Cpu2  :   0.0/0.0     0[                                                  ]
%Cpu3  :   0.0/0.0     0[                                                  ]
%Cpu4  :   0.0/0.0     0[                                                  ]
%Cpu5  :   0.0/0.7     1[|                                                 ]
%Cpu6  :   0.0/0.0     0[                                                  ]
%Cpu7  :   0.7/0.7     1[||                                                ]
%Cpu8  :   7.9/21.9   30[||||||||||||||||||||||||||||||                    ]
%Cpu9  :   0.0/0.0     0[                                                  ]
GiB Mem : 32.4/1.953    [                                                  ]
GiB Swap:  0.0/0.977    [                                                  ]
```

Resource Management and HPA

- Edit deployment for update cpu/memory Limits,request

```
praparns-MacBook-Pro% kubectl set resources deployment/webtest --limits=cpu="1",memory=1Gi --requests=cpu="0.8",memory=800Mi --record
--namespace=webtest-namespace
deployment "webtest" resource requirements updated
```

```
praparns-MacBook-Pro% kubectl describe pods webtest-d6986bb64-bm5q9 --namespace=webtest-namespace
Name:           webtest-d6986bb64-bm5q9
Namespace:      webtest-namespace
Node:           minikube/192.168.99.100
Start Time:     Wed, 21 Mar 2018 00:46:27 +0700
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                pod-template-hash=825426620
                version=1.0
Annotations:    <none>
Status:         Running
IP:             172.17.0.11
Controlled By:  ReplicaSet/webtest-d6986bb64
Containers:
  webtest:
    Container ID:   docker://7e1ae95ed7db8fb9c91b9bcf89b97aedef9e8592d129852930835e744a12b6b6
    Image:          labdocker/cluster:webservicelite_v1
    Image ID:       docker-pullable://labdocker/cluster@sha256:f0f261307a9a0ce8acf317f533e22c3aa438b3a7c5d4c0b5705ce67504326940
    Port:           5000/TCP
    State:          Running
      Started:      Wed, 21 Mar 2018 00:46:28 +0700
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:     1
      memory:  1Gi
    Requests:
      cpu:        800m
      memory:     800Mi
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-n8h4m (ro)
```

Workshop: Resource MNG/HPA

- Part 2: Namespace level configuration

192.168.99.100:32500

Apps | NMac Ked - Mac O... | Medium | jerkins | vagrant | Mesos | Vue | docker | NGINX | Taiwan | Kuberr

Welcome Page from Container Python Lab Web Version 1.00

Checkpoint Date/Time: Sat Jul 8 14:10:26 2017

```
System Info:
 Machine ID:                 88a2186a47ff442aa72465d9e09fc21f
 System UUID:                F631F911-A47C-493D-BFCB-872F1F5C2D30
 Boot ID:                    99b9bb5a-67ba-43d0-bed1-e64fe04c367d
 Kernel Version:             4.9.13
 OS Image:                   Buildroot 2017.02
 Operating System:           linux
 Architecture:               amd64
 Container Runtime Version:  docker://1.12.6
 Kubelet Version:            v1.7.0-alpha.2
 Kube-Proxy Version:         v1.7.0-alpha.2
ExternalID:                  minikube
Non-terminated Pods:         (4 in total)
  Namespace                  Name                                CPU Requests  CPU Limits   Memory Requests  Memory Limits
  ---------                  ----                                ------------  ----------   ---------------  -------------
  kube-system                kube-addon-manager-minikube         5m (0%)       0 (0%)       50Mi (2%)        0 (0%)
  kube-system                kube-dns-268032401-9h5s6            260m (2%)     0 (0%)       110Mi (5%)       170Mi (8%)
  kube-system                kubernetes-dashboard-2vk98          0 (0%)        0 (0%)       0 (0%)           0 (0%)
  webtest-namespace          webtest-4261491039-pft5m            200m (2%)     300m (3%)    100Mi (5%)       200Mi (10%)
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  CPU Requests  CPU Limits    Memory Requests  Memory Limits
  ------------  ----------    ---------------  -------------
  465m (4%)     300m (3%)     260Mi (13%)      370Mi (19%)
```

Page 1

kubernetes by Google

Resource Management and HPA

- Qos for resource management
  - Kubernetes provide 3 type of Qos for defined resource
    - Guaranteed
    - Burstable
    - BestEffort

- Guaranteed: Reserve all resource for support as design, Apply to all init/container equality
  - All Container in the Pod must have a memory limit/request.
  - All Container in the Pod, the memory limit = request.
  - All Container in the Pod must have a CPU limit/request.
  - All Container in the Pod, the CPU limit = request.

- Burstable: Extendable if resource is available
  - Does not meet the criteria for QoS class Guaranteed.
  - At least one Container in the Pod has a memory or CPU request.

- Best Effort:
  - Containers in the Pod must not have any memory or CPU limits or requests.

Workshop: Resource MNG/HPA

- Part 3: Qos POC

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  namespace: qos-namespace
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite
      resources:
        requests:
          memory: "50Mi"
          cpu: "150m"
          ephemeral-storage: "20Mi"
        limits:
          memory: "50Mi"
          cpu: "150m"
          ephemeral-storage: "20Mi"
      ports:
        - containerPort: 5000
          protocol: TCP
```

Resource Management and 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/Memory) and automatic trigger deployment to scale-up application
  - Start from 1.30. We can operate HPA base on container base resource for specification scale by resource of container

kubectl autoscale <type/name> <option:>

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

Resource Management and HPA

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          resources:
            requests:
              cpu: "200m"
          ports:
          - containerPort: 5000
            protocol: TCP
```

```yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  annotations:
  name: webtest
  namespace: default
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: webtest
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 10
#  - type: Resource
#    resource:
#      name: memory
#      target:
#        type: Utilization
#        averageUtilization: 20
```

```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  annotations:
  name: webtest-container-base
  namespace: default
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: webtest
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: ContainerResource
    resource:
      name: cpu
      container: webtest
      target:
        type: Utilization
        averageUtilization: 10
#  - type: ContainerResource
#    resource:
#      container: webtest
#      name: memory
#      target:
#        type: Utilization
#        averageUtilization: 20
```

Resource Management and HPA

Open

Container Resource based Pod Autoscaling #1610

arjunrn opened this issue on Mar 11, 2020 · 72 comments

praparn commented now

As test operate HPAContainerMetrics on Kubernetes 1.30. Just wonder how to make it happen. Base on HPA/V2beta2 was removed since Kubernetes 1.26 and current HPA/V2 is not support "unknown field "spec.containerMetrics", unknown field "spec.metrics[0].resource.container"

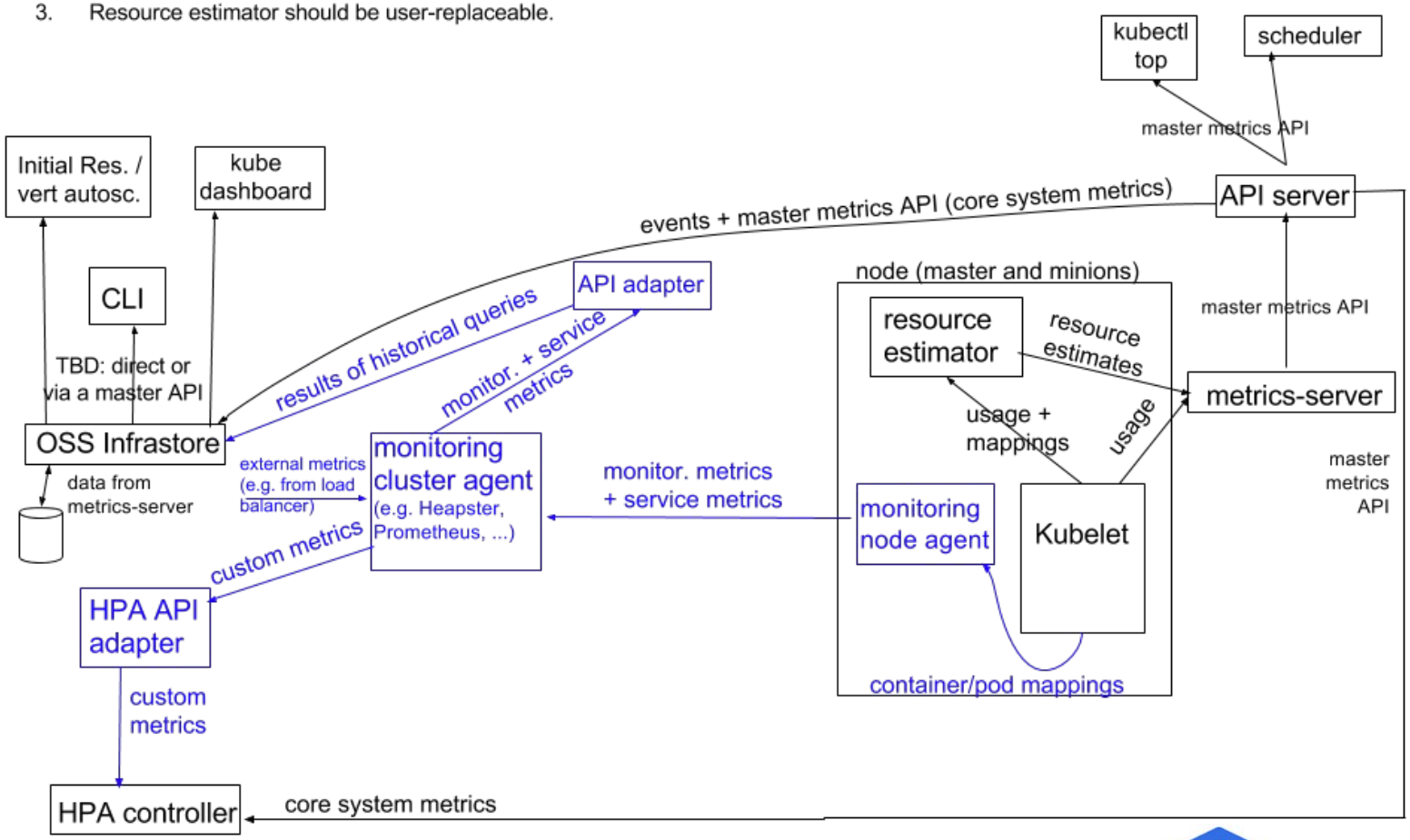
```
ubuntu@ip-10-200-20-103:~/kubernetes_202405/WorkShop_2.8_HPA_WorkShop$ more webtest_hpa_container_base.yml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: webtest-container-base
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: webtest
  minReplicas: 1
  maxReplicas: 10
  containerMetrics:
  - name: webtest # Target container within the Pods
  metrics:
  - type: ContainerResource
    resource:
      name: cpu
      container: webtest
      target:
        type: Utilization
        averageUtilization: 10
#  - type: ContainerResource
#    resource:
#      container: webtest
#      name: memory
#      target:
#        type: Utilization
#        averageUtilization: 10
ubuntu@ip-10-200-20-103:~/kubernetes_202405/WorkShop_2.8_HPA_WorkShop$ kubectl apply -f ./webtest_hpa_container_base.yml
error: resource mapping not found for name: "webtest-container-base" namespace: "" from "./webtest_hpa_container_base.yml": no matches for kind "HorizontalPodAutoscaler" in version "autoscaling/v2beta2"
ensure CRDs are installed first
ubuntu@ip-10-200-20-103:~/kubernetes_202405/WorkShop_2.8_HPA_WorkShop$ kubectl get nodes
NAME                                        STATUS   ROLES           AGE   VERSION
ip-10-200-20-103.ap-southeast-1.compute.internal   Ready    control-plane   72m   v1.30.0
ip-10-200-21-157.ap-southeast-1.compute.internal   Ready    worker          64m   v1.30.0
ip-10-200-22-46.ap-southeast-1.compute.internal    Ready    worker          63m   v1.30.0
ubuntu@ip-10-200-20-103:~/kubernetes_202405/WorkShop_2.8_HPA_WorkShop$
```

Ref: https://github.com/kubernetes/enhancements/issues/1610

Resource Management and HPA

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.



Resource Management and HPA

Algorithm Details

From the most basic perspective, the Horizontal Pod Autoscaler controller operates on the ratio between desired metric value and current metric value:

```
desiredReplicas = ceil[currentReplicas * ( currentMetricValue / desiredMetricValue )]
```

For example, if the current metric value is `200m`, and the desired value is `100m`, the number of replicas will be doubled, since `200.0 / 100.0 == 2.0` If the current value is instead `50m`, we'll halve the number of replicas, since `50.0 / 100.0 == 0.5`. We'll skip scaling if the ratio is sufficiently close to 1.0 (within a globally-configurable tolerance, from the `--horizontal-pod-autoscaler-tolerance` flag, which defaults to 0.1).

--horizontal-pod-autoscaler-initial-readiness-delay

- Default 30 seconds

--horizontal-pod-autoscaler-cpu-initialization-period

- Default 300 seconds

--horizontal-pod-autoscaler-downscale-stabilization

- Default 300 seconds

Resource Management and HPA

- Apply HPA for monitor and scale (TARGET CPU 10%)

```
docker@kubernetes-ms:~$ kubectl autoscale deployment/webtest --min=1 --max=10 --cpu-percent=10
deployment "webtest" autoscaled
docker@kubernetes-ms:~$ kubectl get hpa/webtest
NAME      REFERENCE            TARGETS           MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   <unknown> / 10%   1         10        0          10s
docker@kubernetes-ms:~$ kubectl get hpa/webtest
NAME      REFERENCE            TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   2% / 10%   1         10        1          59s
docker@kubernetes-ms:~$ kubectl describe hpa/webtest
Name:                                                     webtest
Namespace:                                                default
Labels:                                                   <none>
Annotations:                                              <none>
CreationTimestamp:                                        Fri, 02 Feb 2018 10:29:38 -0600
Reference:                                                Deployment/webtest
Metrics:                                                  ( current / target )
  resource cpu on pods  (as a percentage of request):     2% (5m) / 10%
Min replicas:                                             1
Max replicas:                                             10
Conditions:
  Type                  Status  Reason                  Message
  ----                  ------  ------                  -------
  AbleToScale           True    ReadyForNewScale        the last scale time was sufficiently old as to warrant a new scale
  ScalingActive         True    ValidMetricFound        the HPA was able to succesfully calculate a replica count from cpu resource utilization (percent
age of request)
  ScalingLimited        False   DesiredWithinRange      the desired count is within the acceptable range
Events:                 <none>
docker@kubernetes-ms:~$
```

Resource Management and HPA

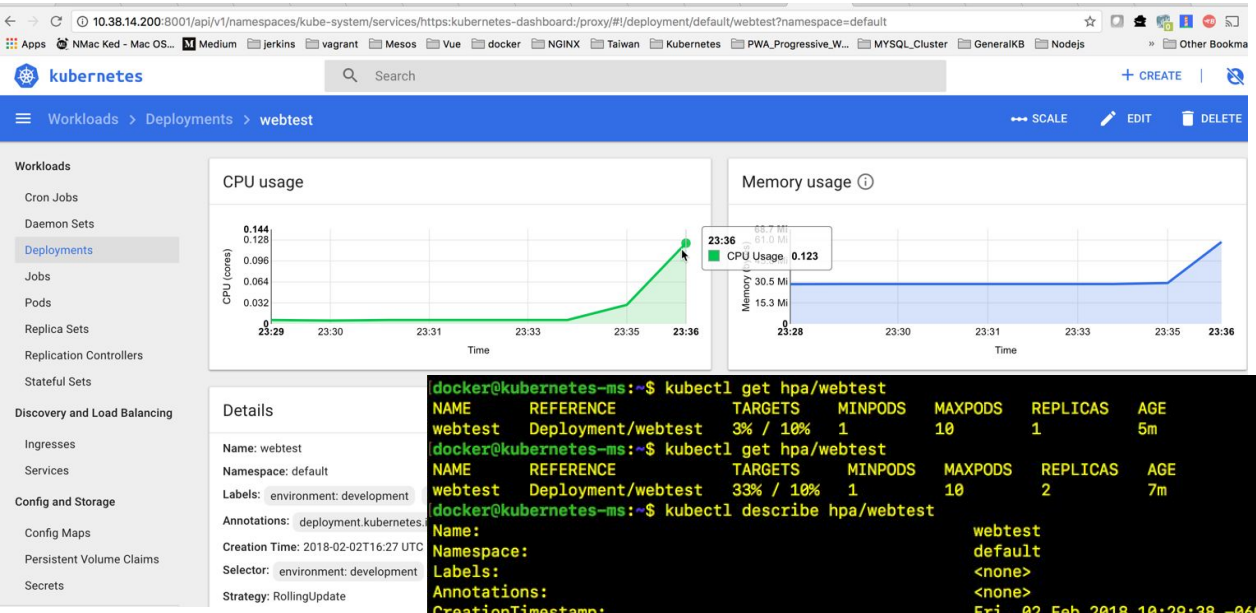
- Generate load by busybox (wget every 10 ms)

```
docker@kubernetes-ms:~$ kubectl run -i --tty load-generator --image=busybox /bin/sh
If you don't see a command prompt, try pressing enter.
/ # while true; sleep 0.01; do wget -q -O- http://webtest.default.svc.cluster.local:5000; done
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:42 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:42 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:34:43 2018
```

```
docker@kubernetes-ms:~$ kubectl top nodes
NAME            CPU(cores)   CPU%      MEMORY(bytes)   MEMORY%
kubernetes-ms   283m         14%       1868Mi          48%
kubernetes-1    83m          4%        1434Mi          37%
kubernetes-2    114m         5%        1410Mi          36%
docker@kubernetes-ms:~$ kubectl top pods
NAME                          CPU(cores)   MEMORY(bytes)
webtest-7d89786977-6zktl      29m          29Mi
docker@kubernetes-ms:~$ kubectl get hpa/webtest
```

Resource Management and HPA

- Load increase and HPA scale-out




```
docker@kubernetes-ms:~$ kubectl get hpa/webtest
NAME      REFERENCE            TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   3% / 10%   1         10        1          5m
docker@kubernetes-ms:~$ kubectl get hpa/webtest
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   33% / 10%   1         10        2          7m
docker@kubernetes-ms:~$ kubectl describe hpa/webtest
Name:                                                     webtest
Namespace:                                                default
Labels:                                                   <none>
Annotations:                                              <none>
CreationTimestamp:                                        Fri, 02 Feb 2018 10:29:38 -0600
Reference:                                                Deployment/webtest
Metrics:                                                  ( current / target )
  resource cpu on pods  (as a percentage of request):     33% (67m) / 10%
Min replicas:                                             1
Max replicas:                                             10
Conditions:
  Type                 Status  Reason                Message
  ----                 ------  ------                -------
  AbleToScale          False   BackoffBoth           the time since the previous scale is still within both the downscale and upscale forbidden windo
ws
  ScalingActive        True    ValidMetricFound      the HPA was able to succesfully calculate a replica count from cpu resource utilization (percent
age of request)
  ScalingLimited       True    ScaleUpLimit          the desired replica count is increasing faster than the maximum scale rate
Events:
  FirstSeen     LastSeen      Count   From                        SubObjectPath   Type          Reason                Message
  ---------     --------      -----   ----                        -------------   --------      ------                -------
  1m            1m            1       horizontal-pod-autoscaler                   Normal        SuccessfulRescale     New size: 2; reason: cpu
 resource utilization (percentage of request) above target
docker@kubernetes-ms:~$
```

Resource Management and HPA

- HPA Scale-Out until meet target (Interval every 5 min)

```
docker@kubernetes-ms:~$ kubectl top pods
NAME                              CPU(cores)   MEMORY(bytes)
webtest-7d89786977-6zktl          69m          29Mi
load-generator-5c4d59d5dd-psqm9   120m         7Mi
webtest-7d89786977-xsp6t          65m          29Mi
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   33% / 10%   1         10        4          10m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   18% / 10%   1         10        4          11m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   18% / 10%   1         10        8          14m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   13% / 10%   1         10        8          15m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   11% / 10%   1         10        8          16m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   11% / 10%   1         10        8          17m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   10% / 10%   1         10        9          19m
```

```
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:12 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:13 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:14 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:14 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Fri Feb  2 16:50:14 2018
```

Resource Management and HPA

- Stop load and HPA Scale-down

```
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
<H1> Welcome Page from Container Python Lab Web Version 1.00 </H1>Checkpoint Date/Time: Tue Mar 20 18:16:52 2018
^C
/ # exit
Session ended, resume using 'kubectl attach load-generator-5c4d59d5dd-d8k24 -c load-generator -i -t' command when the pod is running
praparns-MacBook-Pro% kubectl get deployment
NAME             DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
load-generator   1         1         1            0           12m
webtest          9         9         9            9           19m
praparns-MacBook-Pro% kubectl delete deployment/load-generator
deployment "load-generator" deleted
praparns-MacBook-Pro%
```

Resource Management and HPA

- Stop load and HPA Scale-down

```
docker@kubernetes-ms:~$ kubectl top nodes
NAME            CPU(cores)   CPU%      MEMORY(bytes)   MEMORY%
kubernetes-2    374m         18%       1526Mi          39%
kubernetes-ms   390m         19%       1956Mi          50%
kubernetes-1    184m         9%        1537Mi          39%
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   5% / 10%   1         10        9          22m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   3% / 10%   1         10        3          25m
docker@kubernetes-ms:~$ kubectl get hpa
NAME      REFERENCE            TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
webtest   Deployment/webtest   3% / 10%   1         10        1          30m
docker@kubernetes-ms:~$ kubectl describe hpa/webtest
Name:                                                     webtest
Namespace:                                                default
Labels:                                                   <none>
Annotations:                                              <none>
CreationTimestamp:                                        Fri, 02 Feb 2018 10:29:38 -0600
Reference:                                                Deployment/webtest
Metrics:                                                  ( current / target )
  resource cpu on pods  (as a percentage of request):     3% (6m) / 10%
Min replicas:                                             1
Max replicas:                                             10
Conditions:
  Type                  Status  Reason                  Message
  ----                  ------  ------                  -------
  AbleToScale           False   BackoffBoth             the time since the previous scale is still within both the downscale and upscale forbidden windows
  ScalingActive         True    ValidMetricFound        the HPA was able to succesfully calculate a replica count from cpu resource utilization (percentage of request)
  ScalingLimited        False   DesiredWithinRange      the desired count is within the acceptable range
Events:
  FirstSeen     LastSeen        Count   From                           SubObjectPath   Type           Reason                 Message
  ---------     --------        -----   ----                           -------------   --------       ------                 -------
  24m           24m             1       horizontal-pod-autoscaler                      Normal         SuccessfulRescale      New size: 2; reason: cpu resource utilization
(percentage of request) above target
  21m           21m             1       horizontal-pod-autoscaler                      Normal         SuccessfulRescale      New size: 4; reason: cpu resource utilization
(percentage of request) above target
  17m           17m             1       horizontal-pod-autoscaler                      Normal         SuccessfulRescale      New size: 8; reason: cpu resource utilization
(percentage of request) above target
  13m           13m             1       horizontal-pod-autoscaler                      Normal         SuccessfulRescale      New size: 9; reason: cpu resource utilization
(percentage of request) above target
  7m            7m              1       horizontal-pod-autoscaler                      Normal         SuccessfulRescale      New size: 3; reason: All metrics below target
  1m            1m              1       horizontal-pod-autoscaler                      Normal         SuccessfulRescale      New size: 1; reason: All metrics below target
docker@kubernetes-ms:~$
```

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
- Both of configmap and secret can setup "immutable" for protect accident update value (Start on 1.21)

ConfigMap and Secret

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

kubectl create configmap <name> <option>

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

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

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: game-demo
data:
  # property-like keys; each key maps to a simple value
  player_initial_lives: "3"
  ui_properties_file_name: "user-interface.properties"

  # file-like keys
  game.properties: |
    enemy.types=aliens,monsters
    player.maximum-lives=5
  user-interface.properties: |
    color.good=purple
    color.bad=yellow
    allow.textmode=true
immutable: true
```

ConfigMap and Secret

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

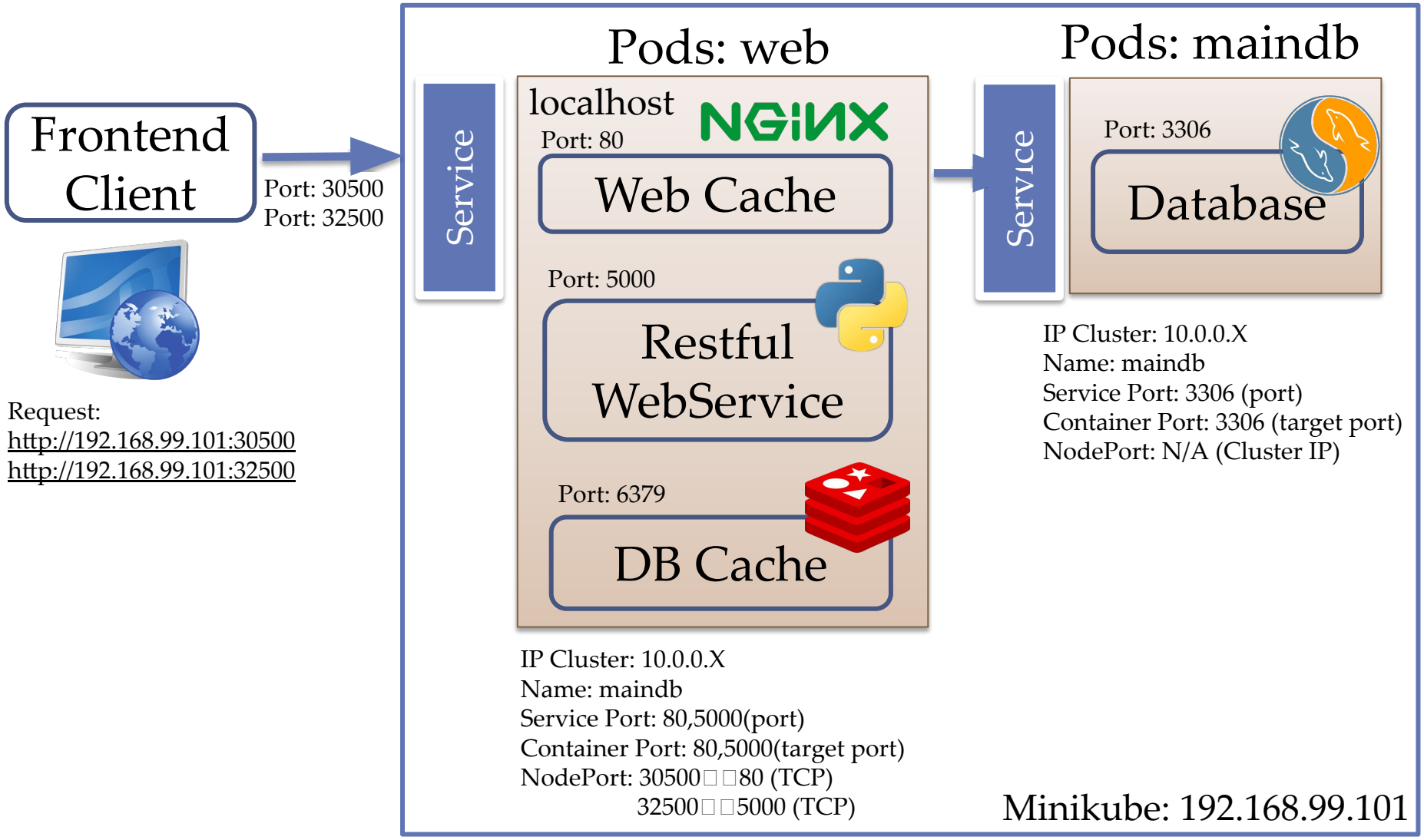
kubectl create secret generic <name> <option>

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

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

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

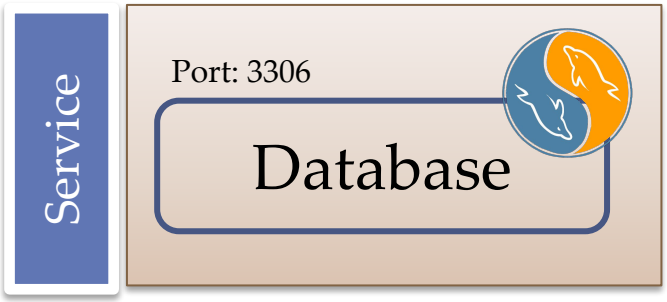
ConfigMap and Secret



Kubernetes: Production Workload Orchestration

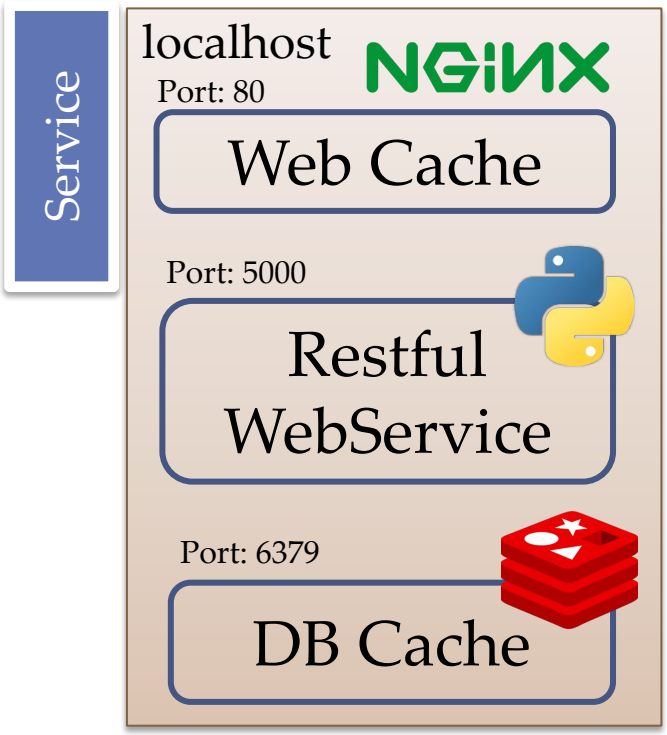
kubernetes by Google

ConfigMap and Secret



Pods.yml: databasemodule_pod.yml

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



Sourcecode: main.py

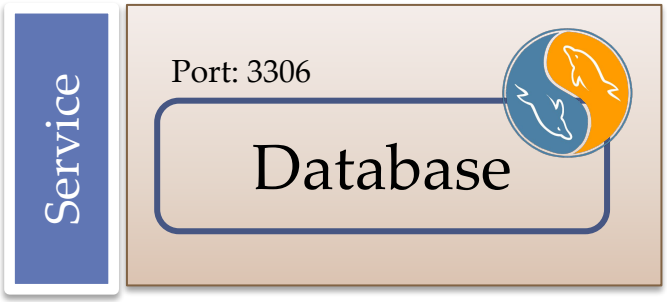
```python
CACHE_DB = redis.Redis(host=os.environ.get('REDIS_HOST', 'cachedb'), port=6379)
db = MySQLdb.connect("maindb","root","password")
MAIN_DB = db.cursor()
```

Pods.yml: webmodule_pod.yml

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


kubernetes by Google

ConfigMap and Secret



Secret.yml: databasemodule_secret.yml

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

Deploy.yml: databasemodule_deploy_config.yml

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

ConfigMap and Secret

Secret.yml:
databasemodule_secret.yml

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

Deploy.yml:
webmodule_deploy_config.yml

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

ConfigMap.yml:
webmodule_configmap.yml

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

Workshop: ConfigMap and Secret

```
~ — Terminal MAC Pro — zsh    Terminal MAC Pro — WinSCP.exe TERM_...    Terminal MAC Pro — ssh ‹ docker-mach...    ~ — Terminal MAC Pro — -bash
    Environment:
      username:                <set to the key 'username' in secret 'databasemodule-secret'>   Optional: false
      MYSQL_ROOT_PASSWORD:     <set to the key 'password' in secret 'databasemodule-secret'>   Optional: false
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-ts1ql (ro)
Conditions:
  Type          Status
  Initialized   True
  Ready         True
  PodScheduled  True
Volumes:
  default-token-ts1ql:
    Type:       Secret (a volume populated by a Secret)
    SecretName: default-token-ts1ql
    Optional:   false
QoS Class:      BestEffort
Node-Selectors: <none>
Tolerations:    <none>
```

```
praparns-MacBook-Pro% kubectl describe configmap/webmodule-configmap --namespace webmicroservice
Name:           webmodule-configmap
Namespace:      webmicroservice
Labels:         environment=development
                module=ConfigMap
                name=webmodule-configmap
                owner=Praparn_L
                version=1.0
Annotations:    <none>

Data
====
REDIS_HOST:
----
localhost
praparns-MacBook-Pro% kubectl describe secret/databasemodule-secret --namespace webmicroservice
Name:           databasemodule-secret
Namespace:      webmicroservice
Labels:         environment=development
                module=Secret
                name=databasemodule-secret
                owner=Praparn_L
                version=1.0
Annotations:    <none>

Type:   Opaque

Data
====
password:       8 bytes
username:       4 bytes
```

Recap Part 1

- Container concept (Recap)
- Introduction to Kubernetes
- System Architecture
- Fundamental of Kubernetes
  - Pods, Container and Services
  - Replication Controller (RC)
  - Deployment/Replica-Set (RS) and Rolling update
  - Volume
  - Liveness and Readyness Probe
  - Resource Management and Horizontal Pods Autoscaling (HPA)
  - ConfigMap Secret

Outline Part 2

- Fundamental of Kubernetes
  - Job and CronJob
  - Log and Monitoring
- Ingress & Gateway API Networking
- Security on Kubernetes
  - Network Policy
  - Volume Policy
  - Resource Usage Policy
  - Resource Consumption Policy
  - Access Control Policy
  - Security Policy
  - Encryption Provider
- Kubernetes in real world
  - Cluster Setup for Bare Metal
  - Orchestrator Assignment
    - nodeSelector
    - Interlude
    - Affinity
    - Taints/Tolerations
- Stateful application deployment
  - Consideration and Awareness
  - Persistent Volumes
  - StatefulSets

Question & Answer Section

By: Praparn L (eva10409@gmail.com)

kubernetes by Google

KhaDev PRESENT

WORKSHOP ADVANCED DOCKER

WITH

kubernetes

สอนการ deploy Dockers บน Kubernetes
จากประสบการณ์ใช้งานจริงบน Production ของ
application ระดับประเทศ

วิทยากร : คุณ PRAPARN LUNGPOONLARP
INFRASTRUCTURE ENGINEER, NETWORK ENGINEER,
SYSTEM ENGINEER

docker kubernetes KhaDev 3DS INTERACTIVE

Part 2

Outline Part 2

- Fundamental of Kubernetes
  - Job and CronJob
  - Log and Monitoring
- Ingress & Gateway API Networking
- Security on Kubernetes
  - Network Policy
  - Volume Policy
  - Resource Usage Policy
  - Resource Consumption Policy
  - Access Control Policy
  - Security Policy
- Kubernetes in real world
  - Cluster Setup for Bare Metal
  - Orchestrator Assignment
    - nodeSelector
    - Interlude
    - Affinity
    - Taints/Tolerations
- Stateful application deployment
  - Consideration and Awareness
  - Persistent Volumes
  - StatefulSets

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
- "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

kubectl create –f <YAML File>

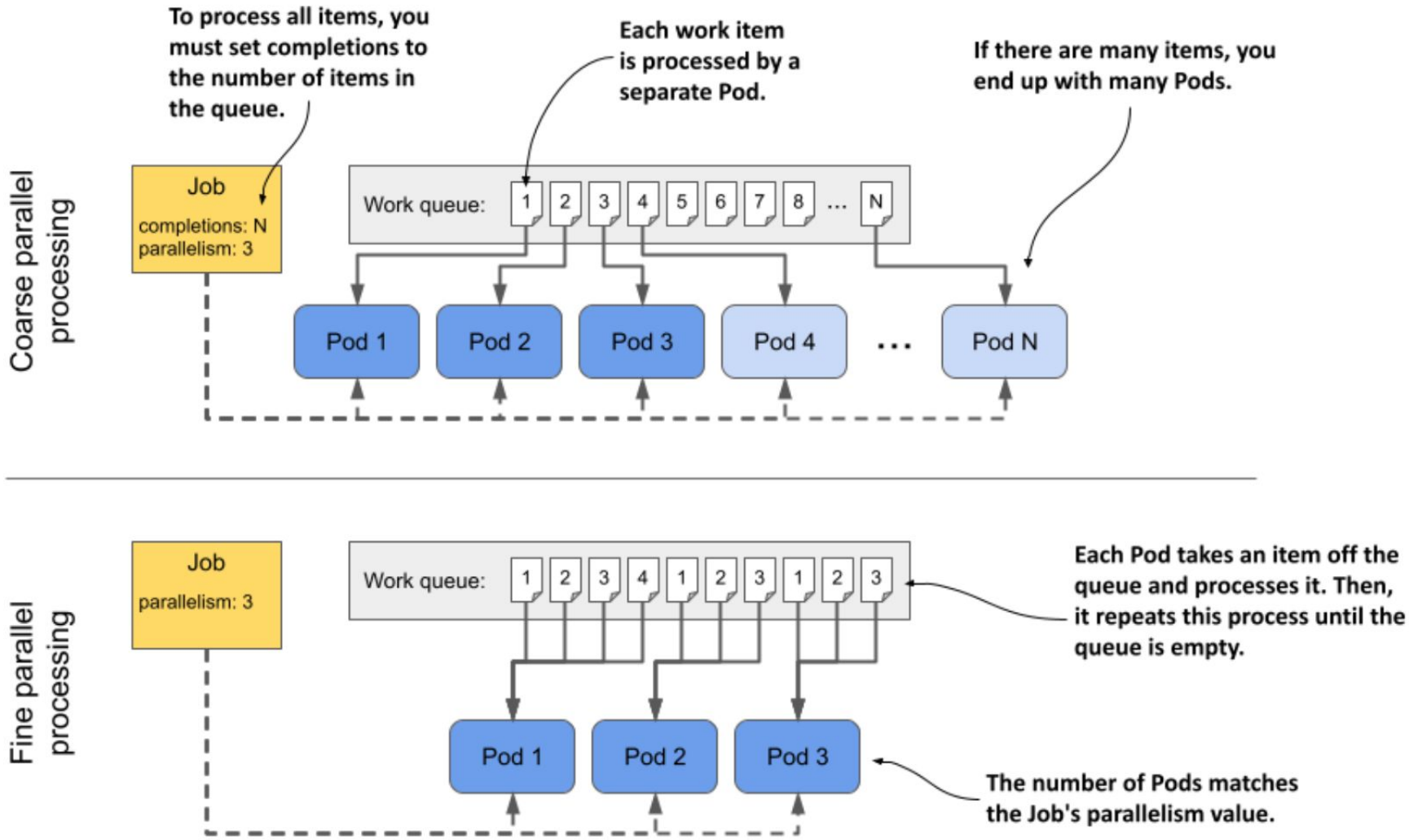
Job and Cron Job

- Parallel execution for Jobs
  - Non-parallel Jobs (Default):
    - Normally, only one Pod is started, unless the Pod fails.
    - The Job is complete as soon as its Pod terminates successfully.
  - Parallel Jobs with a fixed completion count:
    - **“completions: xx”** //How many pods spin at each run
    - The Job represents the overall task, and is complete when there are all successful Pods.
    - **“completionMode="Indexed"** //Define index for pods indentify
  - Parallel Jobs with a work queue:
    - **“parallelism: xx”** //each Pod is independently capable of determining whether or not all its peers are done, and thus that the entire Job is done.
    - the Pods must coordinate amongst themselves or an external service to determine what each should work on.

Job and Cron Job

- Parallel execution for Jobs (Continue)
  - Parallel Jobs with a work queue:
    - When any Pod from the Job terminates with success, no new Pods are created.
    - Once at least one Pod has terminated with success and all Pods are terminated, then the Job is completed with success.
    - Once any Pod has exited with success, no other Pod should still be doing any work for this task or writing any output. They should all be in the process of exiting.

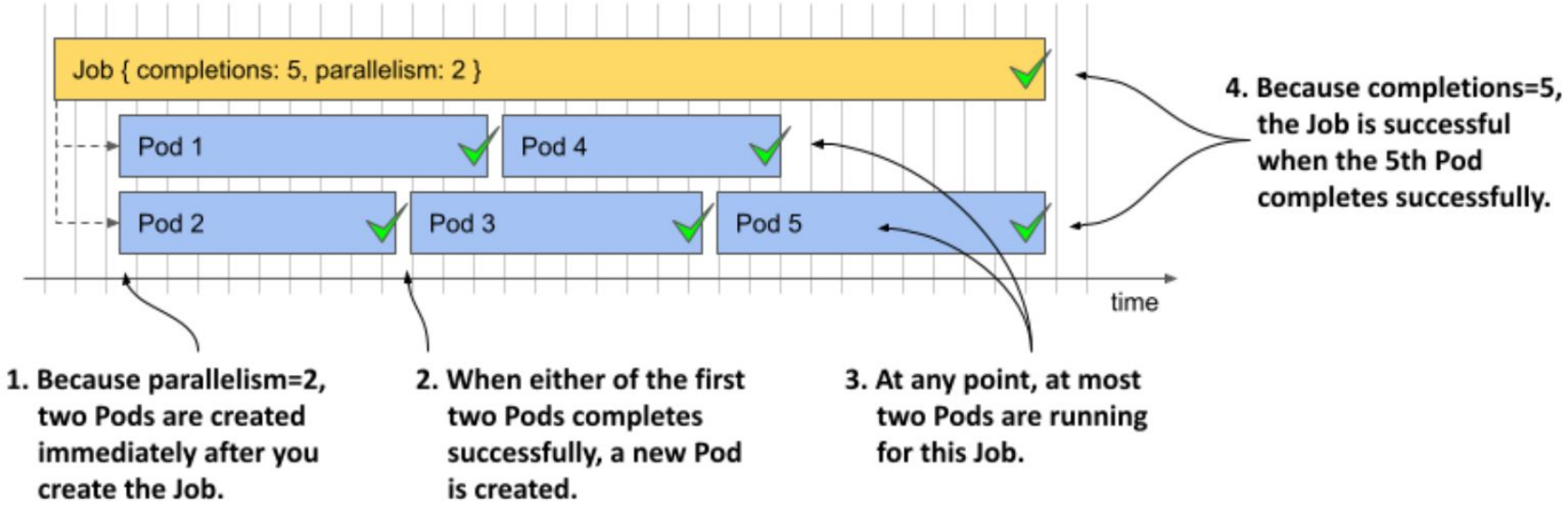
Job and Cron Job



Ref: https://wangwei1237.github.io/Kubernetes-in-Action-Second-Edition/docs/Running_tasks_with_the_Job_resource.html

Job and Cron Job

Figure 17.3 Running a parallel Job with completion=5 and parallelism=2



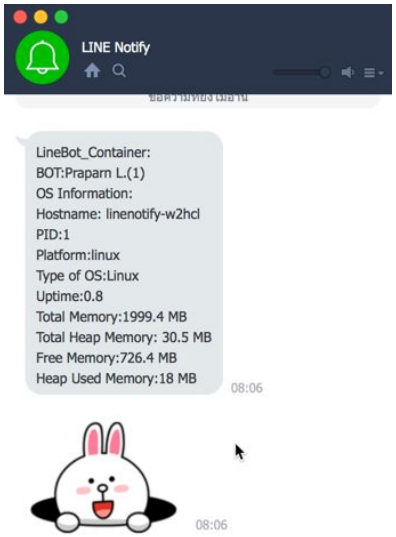
Ref: https://wangwei1237.github.io/Kubernetes-in-Action-Second-Edition/docs/Running_tasks_with_the_Job_resource.html

Job and Cron Job

Job.yml:

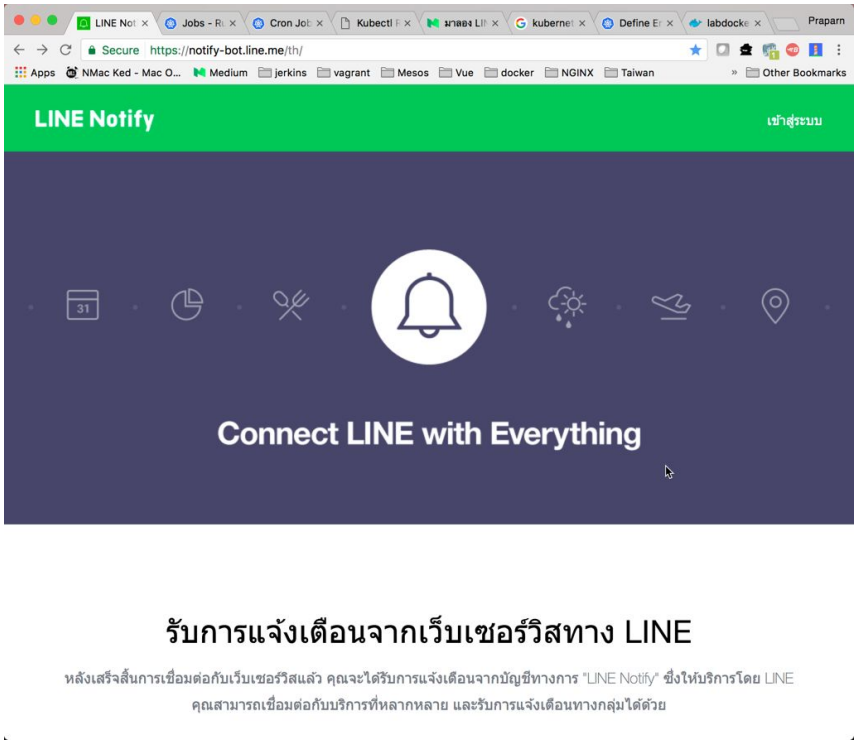
```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: linenotify
  labels:
    name: linenotify
    owner: Praparn_L
    version: "1.0"
    module: Job
    environment: development
spec:
  completions: 1
  parallelism: 1
  Completion Mode:  Indexed
  ttlSecondsAfterFinished: 30
  activeDeadlineSeconds: 60
  template:
    metadata:
      name: linenotify
    spec:
      containers:
      - name: linenotify
        image: labdocker/linenotify:onetime
        env:
        - name: TITLE
          value: "BOT:Praparn L."
        - name: INTERVAL
          value: "10000"
        - name: HEAP_HIGH
          value: "20"
        - name: MEM_HIGH
          value: "20"
        - name: SH_OS
          value: "Y"
        - name: TOKEN
          value: "pKCqcUWGmCpEoNBDNZxsnt5OT4kP41F2BK3QsdSdMDw"
      restartPolicy: Never
```

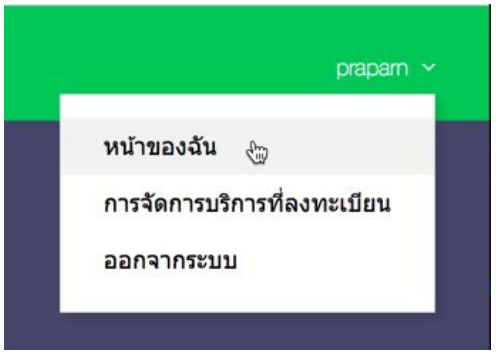
```
praparns-MacBook-Pro% kubectl create -f job.yml
job "linenotify" created
praparns-MacBook-Pro% kubectl get jobs
NAME          DESIRED   SUCCESSFUL   AGE
linenotify    1         0            1m
praparns-MacBook-Pro% kubectl get pods
NAME                 READY     STATUS    RESTARTS   AGE
linenotify-w2hcl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



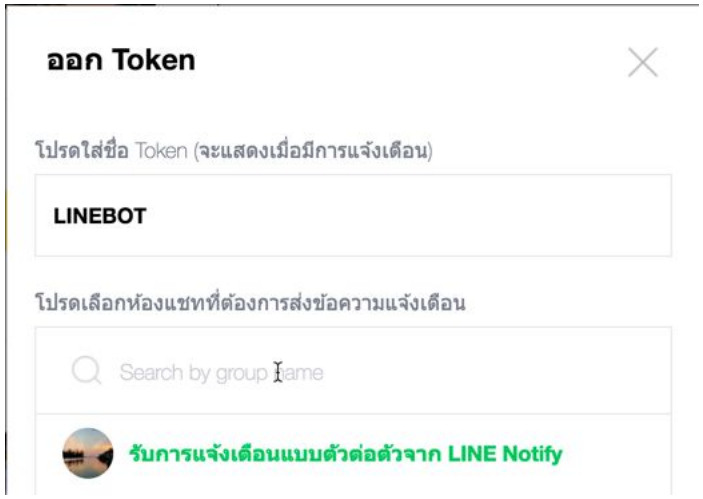
Workshop: Job and CronJob

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





Workshop: Job and CronJob



Workshop: Job and CronJob

- Task1: Create Jobs for Monitor System via LINE

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

LINE

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

kubernetes by Google

Workshop: Job and CronJob

- Task2: Create Cronjob for Monitor System via LINE

WorkShop_1.9_Job_CronJob > ! job_notify_cron.yml

```yaml
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: linenotify
  labels:
    name: linenotify
    owner: Praparn_L
    version: "1.0"
    module: Job
    environment: development
spec:
  schedule: "*/1 * * * *"
  successfulJobsHistoryLimit: 5
  failedJobsHistoryLimit: 1
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: linenotify
              image: labdocker/linenotify:onetime
              env:
              - name: TITLE
                value: "BOT NOTIFY:Praparn L."
              - name: INTERVAL
                value: "10000"
              - name: HEAP_HIGH
                value: "20"
              - name: MEM_HIGH
                value: "20"
              - name: SH_OS
                value: "N"
              - name: TOKEN
                value: "EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX"
          restartPolicy: Never
```

LINE

```
Every 2.0s: kubectl get jobs

NAME                    COMPLETIONS   DURATION   AGE
linenotify-1570292460   1/1           14s        3m15s
linenotify-1570292520   1/1           14s        2m15s
linenotify-1570292580   1/1           15s        75s
linenotify-1570292640   1/1           15s        15s
```


kubernetes by Google

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)
```

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*.

If `startingDeadlineSeconds` is set to a large value or left unset (the default) and if `concurrencyPolicy` is set to `Allow`, the jobs will always run at least once.

For every CronJob, the CronJob Controller checks how many schedules it missed in the duration from its last scheduled time until now. If there are more than 100 missed schedules, then it does not start the job and logs the error

```
Cannot determine if job needs to be started. Too many missed start time (> 100). Set or decrease .spec.startingDeadlineSeconds or check clock skew.
```

It is important to note that if the `startingDeadlineSeconds` field is set (not `nil`), the controller counts how many missed jobs occurred from the value of `startingDeadlineSeconds` until now rather than from the last scheduled time until now. For example, if `startingDeadlineSeconds` is `200`, the controller counts how many missed jobs occurred in the last 200 seconds.

A CronJob is counted as missed if it has failed to be created at its scheduled time. For example, If `concurrencyPolicy` is set to `Forbid` and a CronJob was attempted to be scheduled when there was a previous schedule still running, then it would count as missed.

For example, suppose a CronJob is set to schedule a new Job every one minute beginning at `08:30:00`, and its `startingDeadlineSeconds` field is not set. If the CronJob controller happens to be down from `08:29:00` to `10:21:00`, the job will not start as the number of missed jobs which missed their schedule is greater than 100.

To illustrate this concept further, suppose a CronJob is set to schedule a new Job every one minute beginning at `08:30:00`, and its `startingDeadlineSeconds` is set to 200 seconds. If the CronJob controller happens to be down for the same period as the previous example (`08:29:00` to `10:21:00`,) the Job will still start at 10:22:00. This happens as the controller now checks how many missed schedules happened in the last 200 seconds (ie, 3 missed schedules), rather than from the last scheduled time until now.

The CronJob is only responsible for creating Jobs that match its schedule, and the Job in turn is responsible for the management of the Pods it represents.

Job and Cron Job

- WorkFlow Job

GitHub, Inc. [US] | https://github.com/argoproj/argo

Apps | NMac Ked - Mac... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | Taiwan | NGINX

slack argoproj

Argoproj - Get stuff done with Kubernetes

argo

News

KubeCon 2018 in Seattle was the biggest KubeCon yet with 8000 developers attending. We connected with many existing and new Argoproj users and contributions, and gave away a lot of Argo T-shirts at our booth sponsored by Intuit!

We were also super excited to see KubeCon presentations about Argo by Argo developers, users and partners.

- CI/CD in Light Speed with K8s and Argo CD
  - How Intuit uses Argo CD.
- Automating Research Workflows at BlackRock
  - Why BlackRock created Argo Events and how they use it.
- Machine Learning as Code
  - How Kubeflow uses Argo Workflows as its core workflow engine and Argo CD to declaratively deploy ML pipelines and models.

If you actively use Argo in your organization and your organization would be interested in participating in the Argo Community, please ask a representative to contact saradhi_sreegiriraju@intuit.com for additional information.

What is Argoproj?

Argoproj is a collection of tools for getting work done with Kubernetes.

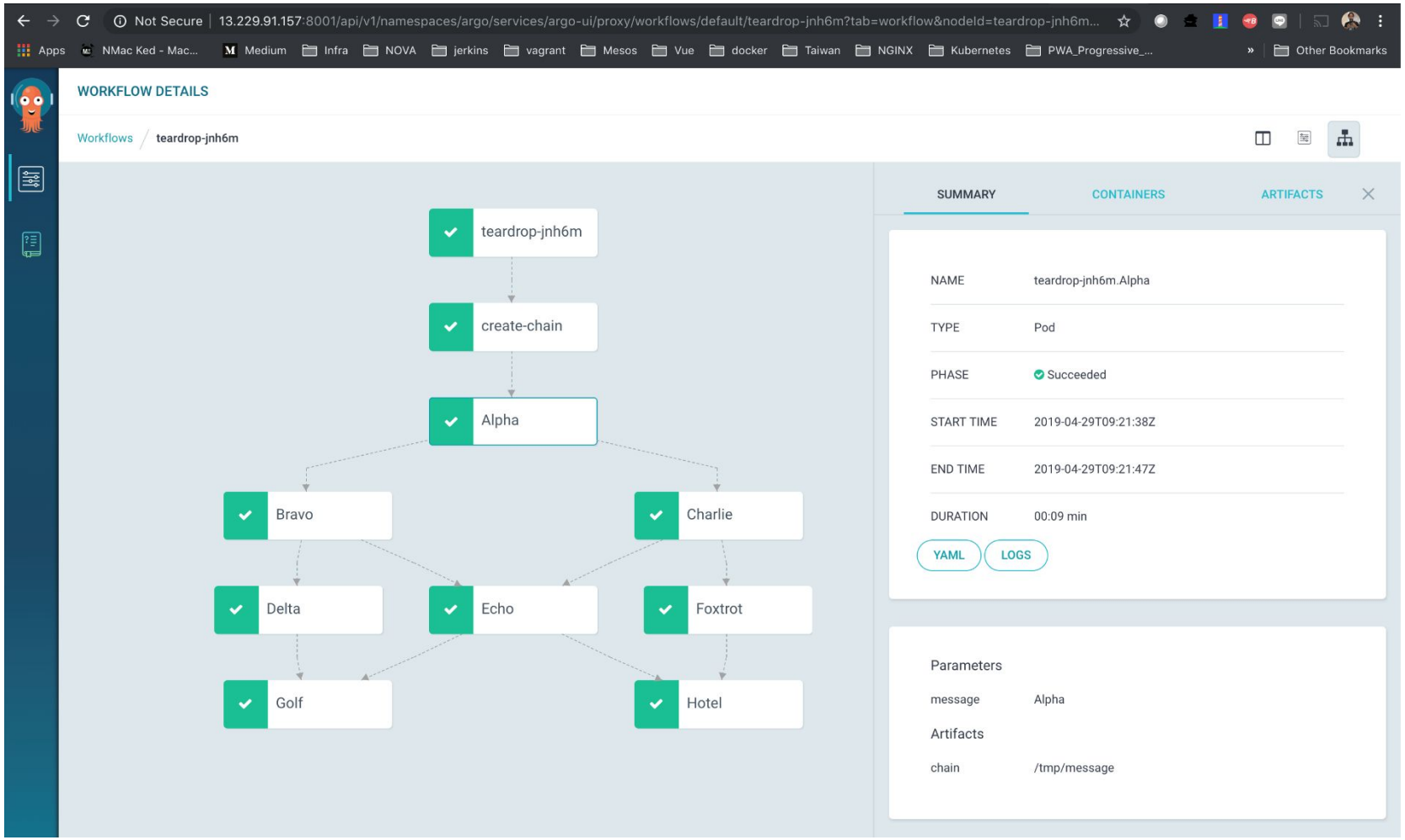
- Argo Workflows - Container-native Workflow Engine
- Argo CD - Declarative GitOps Continuous Delivery
- Argo Events - Event-based Dependency Manager

https://github.com/argoproj/argo

kubernetes by Google

Job and Cron Job

- WorkFlow Job

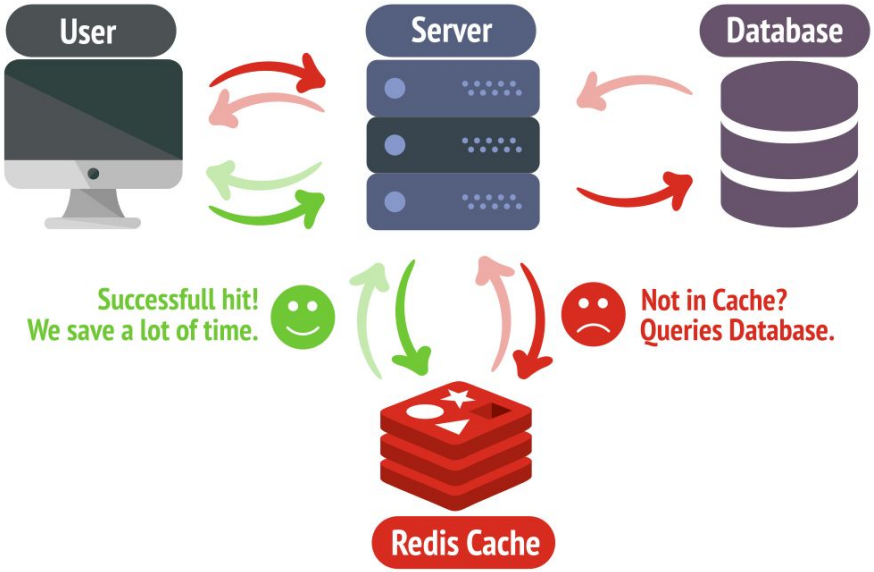


https://github.com/argoproj/argo

Workshop: Job and CronJob

- Task3: Parallel job with Redis

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: job-parallel
spec:
  #completions: 1
  parallelism: 2
  #completionMode: Indexed
  ttlSecondsAfterFinished: 300
  activeDeadlineSeconds: 600
  template:
    metadata:
      name: job-parallel
    spec:
      containers:
      - name: worker
        image: labdocker/redis:worker-test
      restartPolicy: OnFailure
```



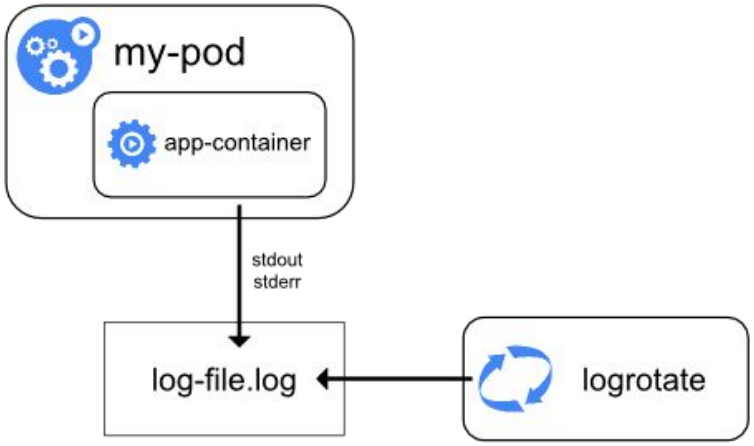
Debug Log and Monitoring

Debug Log and Monitoring

- Normally kubernetes provide tool for check log on Pods and container level with several option

kubectl logs pods/<pods name> -c <container name> <option>

- Option:
  - -p, --previous=false ==> Printout last container fail in Pods
  - -f, --follow=false ==> Follow stream log
  - -c, --container ==> Specific container for check log
  - -l, --selector ==> Select filter some label for check log
- Ex:
  - kubectl logs -f pods/maindb –c maindb



Debug Log and Monitoring

- Log Rotation: (Start on Version 1.21)
  - Tuning kubelet engine for operate log rotation with following parameter
    - containerLogMaxFiles: For maximum number of logs
    - containerLogMaxSize: For maximum size of each log

```yaml
---
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
containerLogMaxFiles: 5    # Maximum number of container logs to retain.
containerLogMaxSize: 1Mi   # Change the size of /var/log/containers/<pod-name>/log files size to 1M max.
maxPods: 110
serializeImagePulls: false
---
```

Ref: https://kubernetes.io/docs/reference/config-api/kubelet-config.v1beta1/

Debug Log and Monitoring

- Monitoring kubernetes system via dashboard

Kubernetes Dashboard

go report A+ | codecov 37% | License Apache 2.0

Introduction

Kubernetes Dashboard is a general purpose, web-based UI for Kubernetes clusters. It allows users to manage applications running in the cluster and troubleshoot them, as well as manage the cluster itself.

As of version 7.0.0, we have dropped support for Manifest-based installation. Only Helm-based installation is supported now. Due to multi-container setup and hard dependency on Kong gateway API proxy it would not be feasible to easily support Manifest-based installation.

Additionally, we have changed the versioning scheme and dropped `appVersion` from Helm chart. It is because, with a multi-container setup, every module is now versioned separately. Helm chart version can be considered an app version now.

Debug Log and Monitoring

- Monitoring kubernetes system via dashboard

localhost

kubernetes | default | Search

Cluster > Nodes > kubernetes-dashboard-control-plane

Local dev environment

Workloads
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

Service
- Ingresses
- Ingress Classes
- Services

Config and Storage
- Config Maps
- Persistent Volume Claims
- Secrets
- Storage Classes

Cluster
- Cluster Role Bindings
- Cluster Roles

CPU

9.4% Requests — Cores: 0,75

1.3% Limits — Cores: 0,1

Memory

2.5% Requests — MiB: 150

0.8% Limits — MiB: 50

Pods

6.4% Allocation — Pods: 7

Conditions

Type	Status	Last probe time	Last transition time	Reason	Message
MemoryPressure	False	2 minutes ago	10 days ago	KubeletHasSufficientMemory	kubelet has sufficient memory available
DiskPressure	False	2 minutes ago	10 days ago	KubeletHasNoDiskPressure	kubelet has no disk pressure
PIDPressure	False	2 minutes ago	10 days ago	KubeletHasSufficientPID	kubelet has sufficient PID available
Ready	True	2 minutes ago	10 days ago	KubeletReady	kubelet is posting ready status

Pods

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
kindnet-fmj8d	docker.io/kindest/kindnetd:v20230511-dc714da8	app: kindnet controller-revision-hash: 58f5b657b8 k8s-app: kindnet	kubernetes-dashboard-control-plane	Running	5	-	-	10 days ago

kubernetes by Google

Debug Log and Monitoring

- Ephemeral Container (GA on 1.25):
  - Getting debug application inside "Pods"
  - Ideally is run container inside pods for operate as "debugger"

FEATURE STATE: `Kubernetes v1.25 [stable]`

This page provides an overview of ephemeral containers: a special type of container that runs temporarily in an existing Pod to accomplish user-initiated actions such as troubleshooting. You use ephemeral containers to inspect services rather than to build applications.

Understanding ephemeral containers

Pods are the fundamental building block of Kubernetes applications. Since Pods are intended to be disposable and replaceable, you cannot add a container to a Pod once it has been created. Instead, you usually delete and replace Pods in a controlled fashion using deployments.

Sometimes it's necessary to inspect the state of an existing Pod, however, for example to troubleshoot a hard-to-reproduce bug. In these cases you can run an ephemeral container in an existing Pod to inspect its state and run arbitrary commands.

What is an ephemeral container?

Ephemeral containers differ from other containers in that they lack guarantees for resources or execution, and they will never be automatically restarted, so they are not appropriate for building applications. Ephemeral containers are described using the same `ContainerSpec` as regular containers, but many fields are incompatible and disallowed for ephemeral containers.

- Ephemeral containers may not have ports, so fields such as `ports`, `livenessProbe`, `readinessProbe` are disallowed.
- Pod resource allocations are immutable, so setting `resources` is disallowed.
- For a complete list of allowed fields, see the EphemeralContainer reference documentation.

Ephemeral containers are created using a special `ephemeralcontainers` handler in the API rather than by adding them directly to `pod.spec`, so it's not possible to add an ephemeral container using `kubectl edit`.

Like regular containers, you may not change or remove an ephemeral container after you have added it to a Pod.

Uses for ephemeral containers

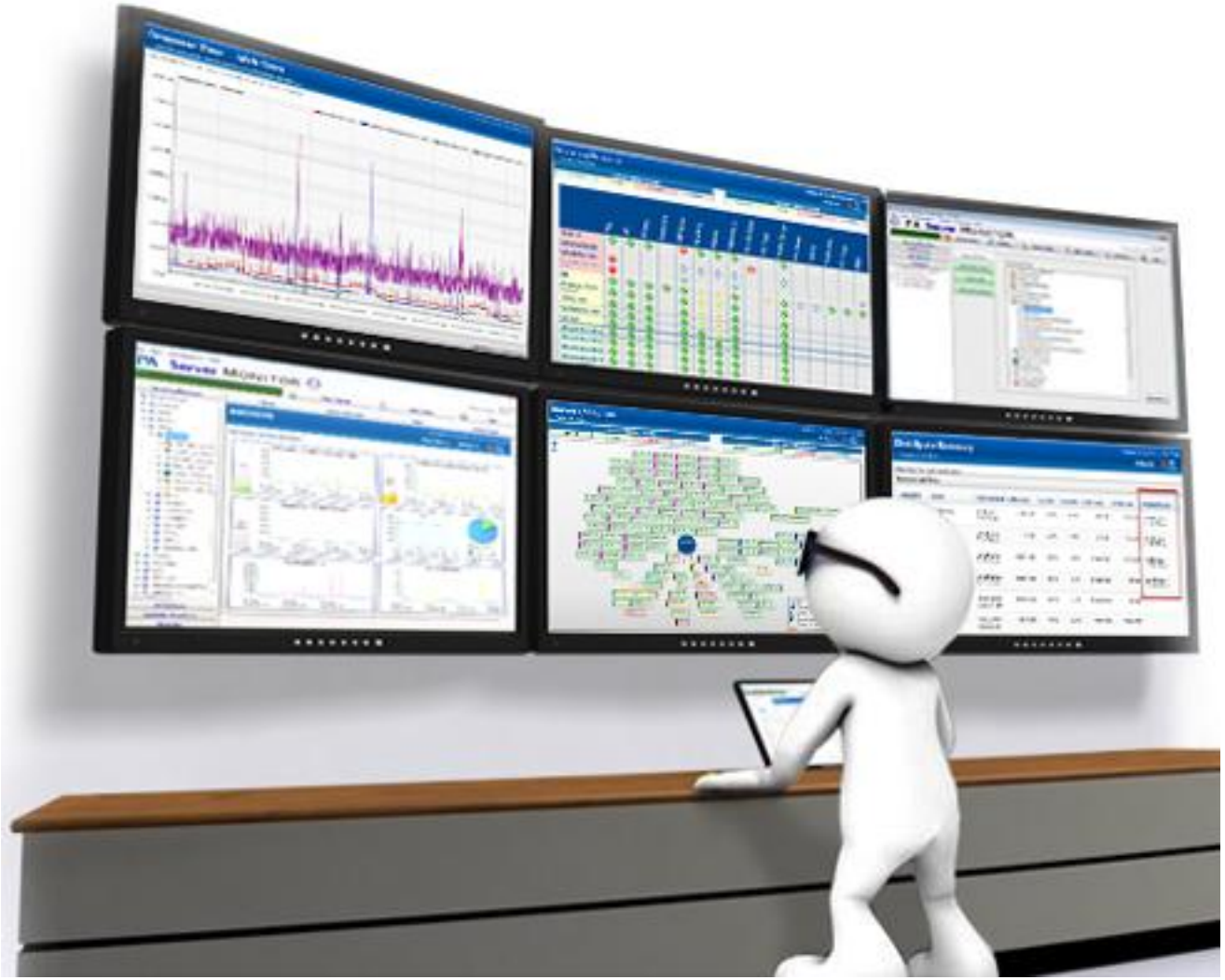
Ephemeral containers are useful for interactive troubleshooting when `kubectl exec` is insufficient because a container has crashed or a container image doesn't include debugging utilities.

kubernetes by Google

Debug Log and Monitoring

```
ubuntu@ip-10-200-20-139:~$ kubectl describe pods/maindb-67c6789d9c-q5p9k
Name:             maindb-67c6789d9c-q5p9k
Namespace:        default
Priority:         0
Service Account:  default
Node:             ip-10-200-20-139/10.200.20.139
Start Time:       Sun, 20 Nov 2022 10:16:45 +0700
Labels:           environment=development
                  module=maindb
                  name=maindb
                  owner=Praparn_L
                  pod-template-hash=67c6789d9c
                  version=1.0
Annotations:      <none>
Status:           Running
IP:               10.0.0.201
IPs:
  IP:           10.0.0.201
Controlled By:  ReplicaSet/maindb-67c6789d9c
Containers:
  maindb:
    Container ID:   containerd://3ff8fc37d0b084414b1d367607cb94dea585ed60b53268297f8efb0c22dd2384
    Image:          labdocker/mariadb:latest
    Image ID:       docker.io/labdocker/mariadb@sha256:03fb19fa5729856ec8c8ed23d421ed1ab6c0e2d63fdf2b1bd8d311025e228a9b
    Port:           3306/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Sun, 20 Nov 2022 10:16:48 +0700
    Ready:          True
    Restart Count:  0
    Environment:
      MYSQL_ROOT_PASSWORD:  password
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-sf7g4 (ro)
Ephemeral Containers:
  debugger-vkmwv:
    Container ID:   containerd://7ca057bf539f1801b00d581edae84f114db7d47cf443a49d09ac14b937a76b5d
    Image:          labdocker/alpine:3.13.6
    Image ID:       docker.io/labdocker/alpine@sha256:e15947432b813e8ffa90165da919953e2ce850bef511a0ad1287d7cb86de84b5
    Port:           <none>
    Host Port:      <none>
    State:          Terminated
      Reason:       Completed
      Exit Code:    0
      Started:      Sun, 20 Nov 2022 10:43:41 +0700
      Finished:     Sun, 20 Nov 2022 10:46:38 +0700
    Ready:          False
    Restart Count:  0
    Environment:    <none>
    Mounts:         <none>
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
```

Workshop: Log and Monitoring



Kubernetes: Production Workload Orchestration

kubernetes by Google

Ingress Network

Ingress Network

- Remember Service ?
  - Service will expose for other Pods / External to connect and access service on Pods inside

```
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$
```

  - Service quite limit and non flexible for operate
    - How to handle for multiple service on same port ?
    - How to limit protocol for access ? (HTTP/HTTPS/FTP etc)
    - How to blinding with hostname ? (www.xxxx.yyy)
    - How to TLS connection ?
    - etc

- Ingress is tool will operate for that
  - Give some customize label for classify host and define some selector criteria for specific node run Pods
  - Ingress can be implemented by different controller and specify with ingress class (categories)

Ingress Network

Kubernetes as a project supports and maintains AWS, GCE, and nginx ingress controllers.

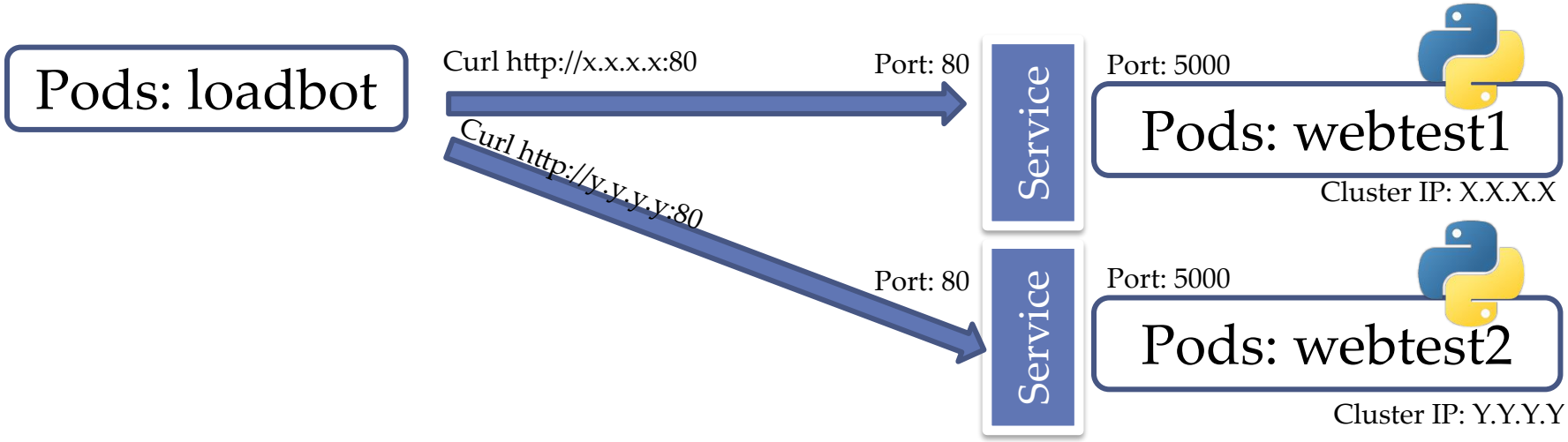
Additional controllers

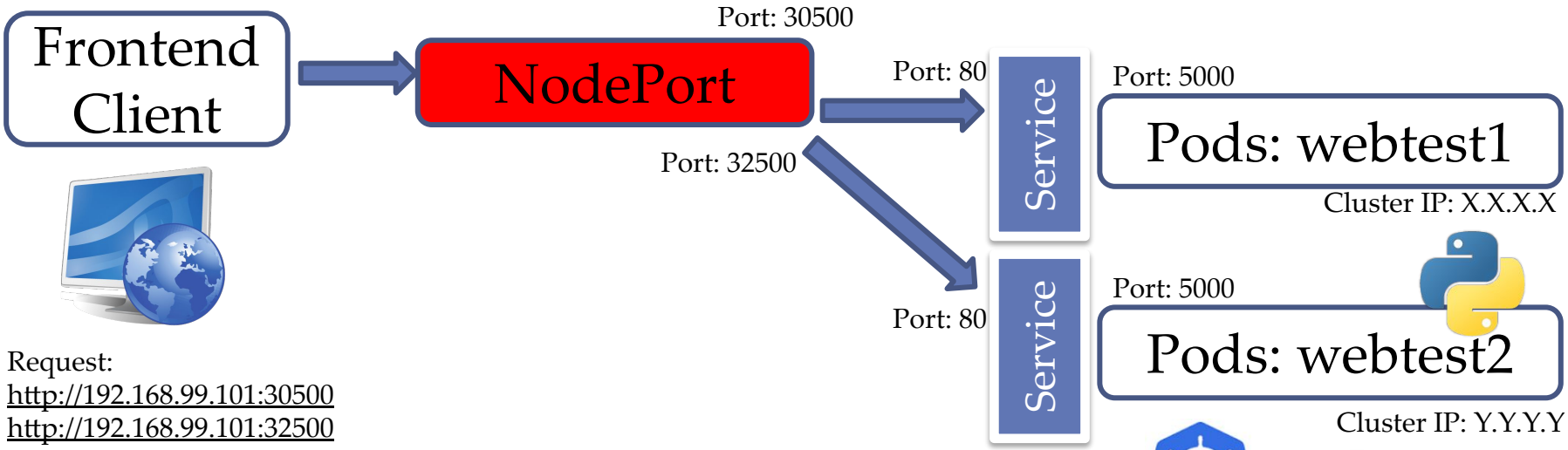
> **Caution:** This section links to third party projects that provide functionality required by Kubernetes. The Kubernetes project authors aren't responsible for these projects. This page follows CNCF website guidelines by listing projects alphabetically. To add a project to this list, read the content guide before submitting a change.

- AKS Application Gateway Ingress Controller is an ingress controller that configures the Azure Application Gateway.
- Ambassador API Gateway is an Envoy-based ingress controller.
- Apache APISIX ingress controller is an Apache APISIX-based ingress controller.
- Avi Kubernetes Operator provides L4-L7 load-balancing using VMware NSX Advanced Load Balancer.
- The Citrix ingress controller works with Citrix Application Delivery Controller.
- Contour is an Envoy based ingress controller.
- EnRoute is an Envoy based API gateway that can run as an ingress controller.
- Easegress IngressController is an Easegress based API gateway that can run as an ingress controller.
- F5 BIG-IP Container Ingress Services for Kubernetes lets you use an Ingress to configure F5 BIG-IP virtual servers.
- Gloo is an open-source ingress controller based on Envoy, which offers API gateway functionality.
- HAProxy Ingress is an ingress controller for HAProxy.
- The HAProxy Ingress Controller for Kubernetes is also an ingress controller for HAProxy.
- Istio Ingress is an Istio based ingress controller.
- The Kong Ingress Controller for Kubernetes is an ingress controller driving Kong Gateway.
- The NGINX Ingress Controller for Kubernetes works with the NGINX webserver (as a proxy).
- Skipper HTTP router and reverse proxy for service composition, including use cases like Kubernetes Ingress, designed as a library to build your custom proxy.
- The Traefik Kubernetes Ingress provider is an ingress controller for the Traefik proxy.
- Tyk Operator extends Ingress with Custom Resources to bring API Management capabilities to Ingress. Tyk Operator works with the Open Source Tyk Gateway & Tyk Cloud control plane.
- Voyager is an ingress controller for HAProxy.

Ingress Network

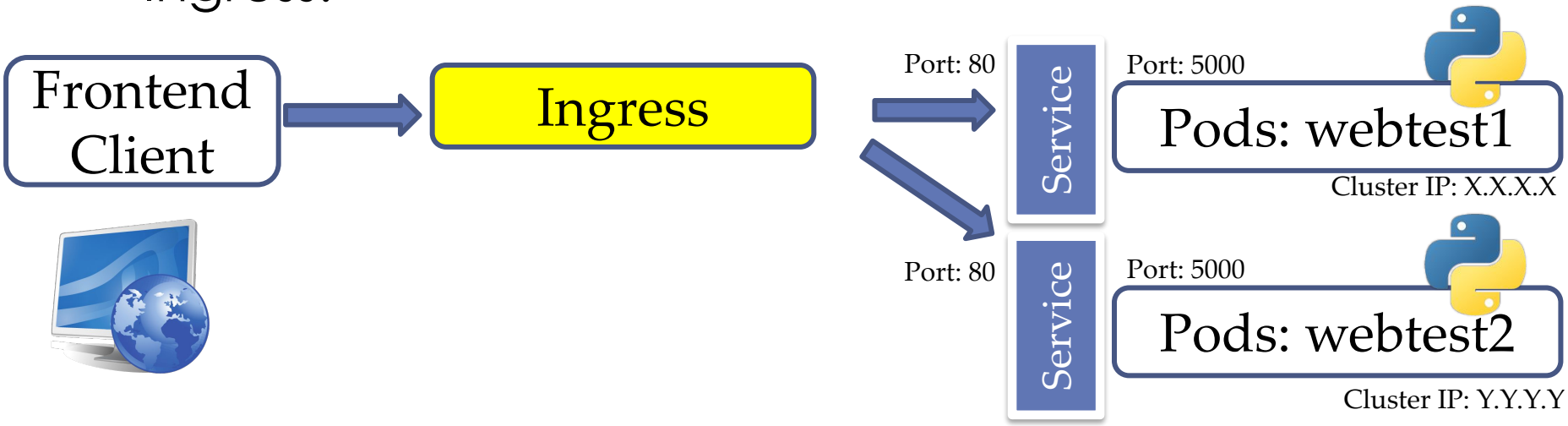
- Service:





Ingress Network

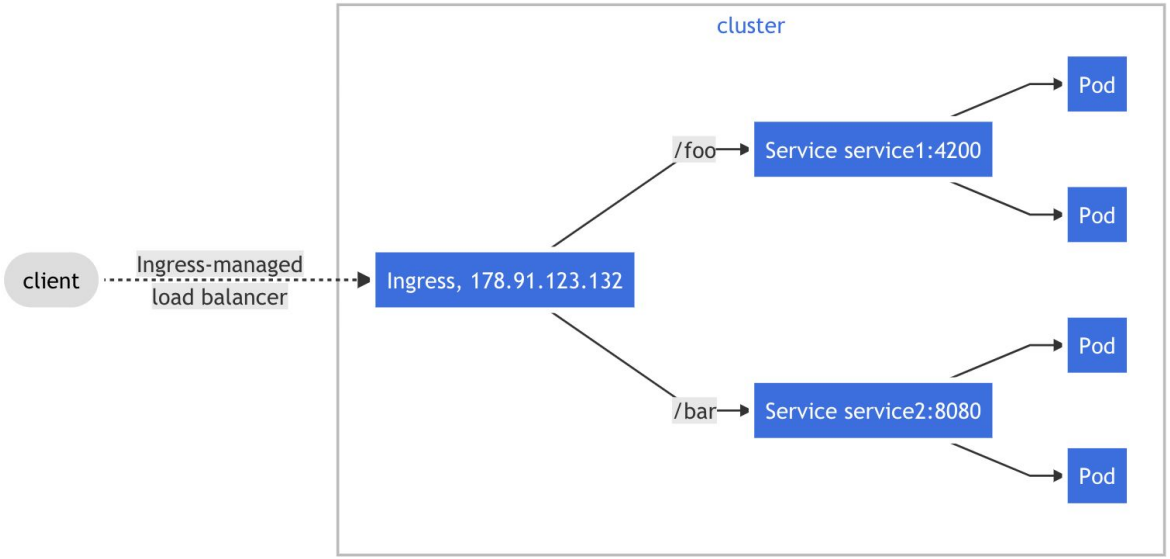
- Ingress:



Ingress Network

Users > praparnlueangphoonlap > Desktop >

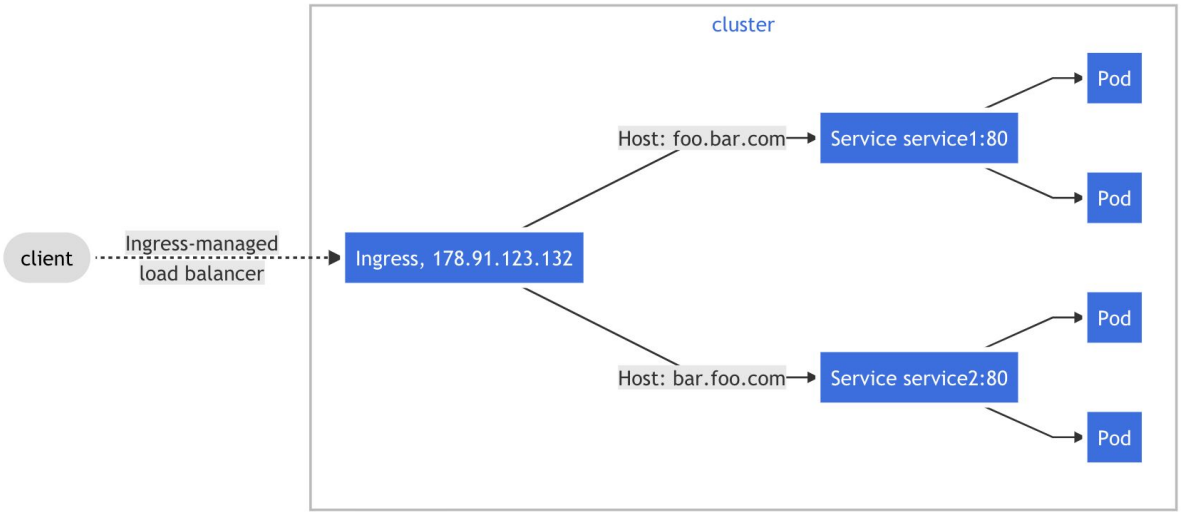
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: simple-fanout-example
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - path: /foo
        pathType: Prefix
        backend:
          service:
            name: service1
            port:
              number: 4200
      - path: /bar
        pathType: Prefix
        backend:
          service:
            name: service2
            port:
              number: 8080
```



Ingress Network

Users > praparnlueangphoonlap > Desktop > !

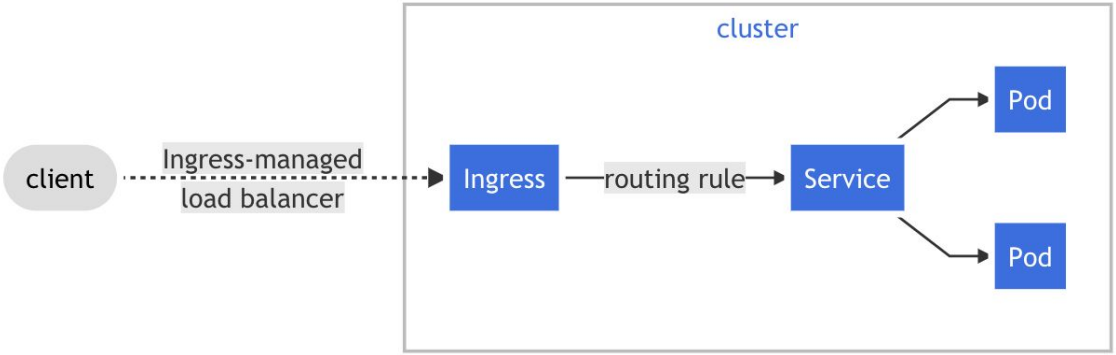
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: name-virtual-host-ingress
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: service1
            port:
              number: 80
  - host: bar.foo.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: service2
            port:
              number: 80
```



Ingress Network

Users > praparnlueangphoonlap > Desktop > ! 1.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: name-virtual-host-ingress-no-third-host
spec:
  rules:
  - host: first.bar.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: service1
            port:
              number: 80
  - host: second.bar.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: service2
            port:
              number: 80
  - http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: service3
            port:
              number: 80
```



Ingress Network

SVC: webtest1

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest1
  labels:
    name: webtest1
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: webtest1
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  ports:
   - port: 80
     name: http
     targetPort: 5000
     protocol: TCP
---
```

SVC: webtest2

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest2
  labels:
    name: webtest2
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: webtest2
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  ports:
   - port: 80
     name: http
     targetPort: 5000
     protocol: TCP
```

Ingress: ingresswebtest

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingresswebtest
spec:
  ingressClassName: nginx
  rules:
  - host: webtest1.kubernetes labthailand.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: webtest1
            port:
              number: 80
  - host: webtest2.kubernetes labthailand.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: webtest2
            port:
              number: 80
```

Ingress Network

- Path Type:
  - Prefix: Match based on URL path (Case sensitive)
  - Exact: Match path exactly and with case sensitivity
  - ImplementationSpecific: Use for ingress class

- Multiple match:
  - Longest match path (1)
  - Exact match path (2) (Exact > Prefix)

Ingress Network

Kind	Path(s)	Request path(s)	Matches?
Prefix	`/`	(all paths)	Yes
Exact	`/foo`	`/foo`	Yes
Exact	`/foo`	`/bar`	No
Exact	`/foo`	`/foo/`	No
Exact	`/foo/`	`/foo`	No
Prefix	`/foo`	`/foo`, `/foo/`	Yes
Prefix	`/foo/`	`/foo`, `/foo/`	Yes
Prefix	`/aaa/bb`	`/aaa/bbb`	No
Prefix	`/aaa/bbb`	`/aaa/bbb`	Yes
Prefix	`/aaa/bbb/`	`/aaa/bbb`	Yes, ignores trailing slash
Prefix	`/aaa/bbb`	`/aaa/bbb/`	Yes, matches trailing slash
Prefix	`/aaa/bbb`	`/aaa/bbb/ccc`	Yes, matches subpath
Prefix	`/aaa/bbb`	`/aaa/bbbxyz`	No, does not match string prefix
Prefix	`/`, `/aaa`	`/aaa/ccc`	Yes, matches `/aaa` prefix
Prefix	`/`, `/aaa`, `/aaa/bbb`	`/aaa/bbb`	Yes, matches `/aaa/bbb` prefix
Prefix	`/`, `/aaa`, `/aaa/bbb`	`/ccc`	Yes, matches `/` prefix
Prefix	`/aaa`	`/ccc`	No, uses default backend
Mixed	`/foo` (Prefix), `/foo` (Exact)	`/foo`	Yes, prefers Exact

Ingress Network

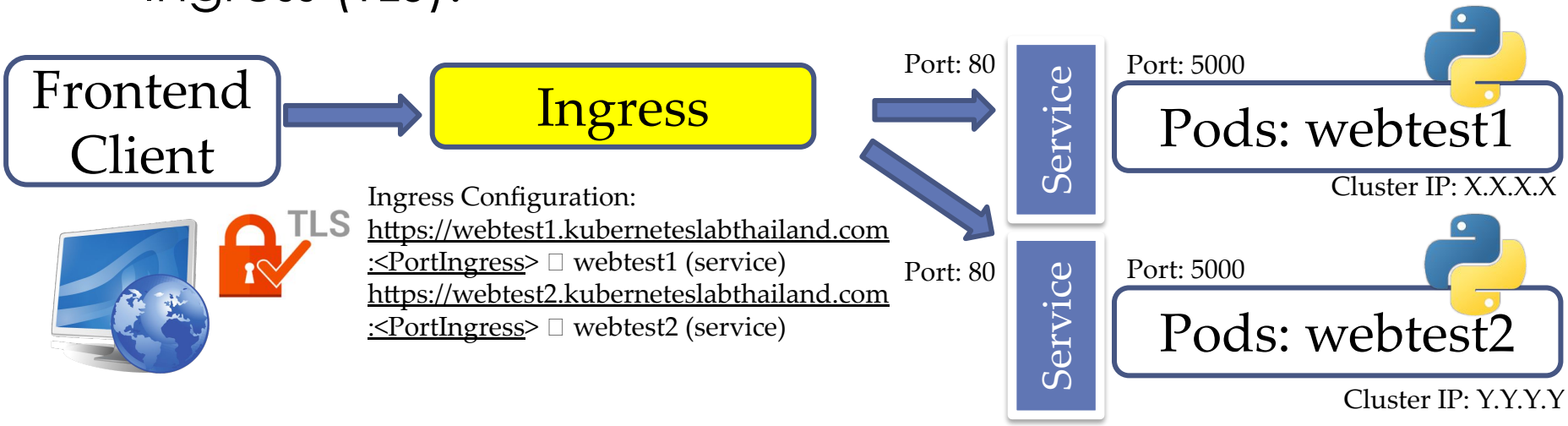
- Multiple-controller of ingress
  - Kubernetes have capability for housing multiple ingress controller on same cluster
  - Each ingress controller

- Ingress Class: Categories of ingress controller

```yaml
---
# Source: ingress-nginx/templates/controller-ingressclass.yaml
# We don't support namespaced ingressClass yet
# So a ClusterRole and a ClusterRoleBinding is required
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  labels:
    helm.sh/chart: ingress-nginx-4.0.1
    app.kubernetes.io/name: ingress-nginx
    app.kubernetes.io/instance: ingress-nginx
    app.kubernetes.io/version: 1.0.0
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/component: controller
  name: nginx
  namespace: ingress-nginx
  #annotation:
  #  ingressclass.kubernetes.io/is-default-class: true  #This for set default ingress controller on cluster
spec:
  controller: k8s.io/ingress-nginx
---
```

Ingress Network

- Ingress (TLS):



Ingress Network

www.selfsignedcertificate.com

NMac Ked - Mac O... | Medium | jerkins | vagrant | Mesos | Vue | docker | NGINX | Taiwan | Kubernetes | PWA_Progressive_... | MYSQL_Cluster | GeneralKB | Nodejs

Self-Signed Certificate Generator | Development Tips | About

Self-Signed Certificate Generator

Self-signed ssl certificates can be used to set up temporary ssl servers. You can use it for test and development servers where security is not a big concern. Use the form below to generate a self-signed ssl certificate and key.

Server name: Generate »

About SSL Certificates

SSL certificates are required in order to run web sites using the HTTPS protocol. For professional web sites, you usually buy such a certificate from Verisign, Thawte or any other ssl certificate vendor. SSL certificates use a chain of trust, where each certificate is signed (trusted) by a higher, more credible certificate. At the top of the chain of trust are the root certificates, owned by Verisign and others. These certificates are typically shipped with your operating system or web browser.

sslstore
AFFORDABLE SSL CERTIFICATES
ISSUED IN MINUTES*
FROM $10.99
SAVE 80%

Ingress Network

SVC: webtest1

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest1
  labels:
    name: webtest1
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: webtest1
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  ports:
   - port: 80
     name: http
     targetPort: 5000
     protocol: TCP
---
```

SVC: webtest2

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest2
  labels:
    name: webtest2
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: webtest2
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  ports:
   - port: 80
     name: http
     targetPort: 5000
     protocol: TCP
```

Ingress: ingresswebtest

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingresswebtesttls
spec:
  ingressClassName: nginx
  tls:
  - hosts:
    - webtest1.kubernetes labthailand.com
    secretName: webtest1secret
  - hosts:
    - webtest2.kubernetes labthailand.com
    secretName: webtest2secret
  rules:
  - host: webtest1.kubernetes labthailand.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: webtest1
            port:
              number: 80
  - host: webtest2.kubernetes labthailand.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: webtest2
            port:
              number: 80
```

Secret: webtest1

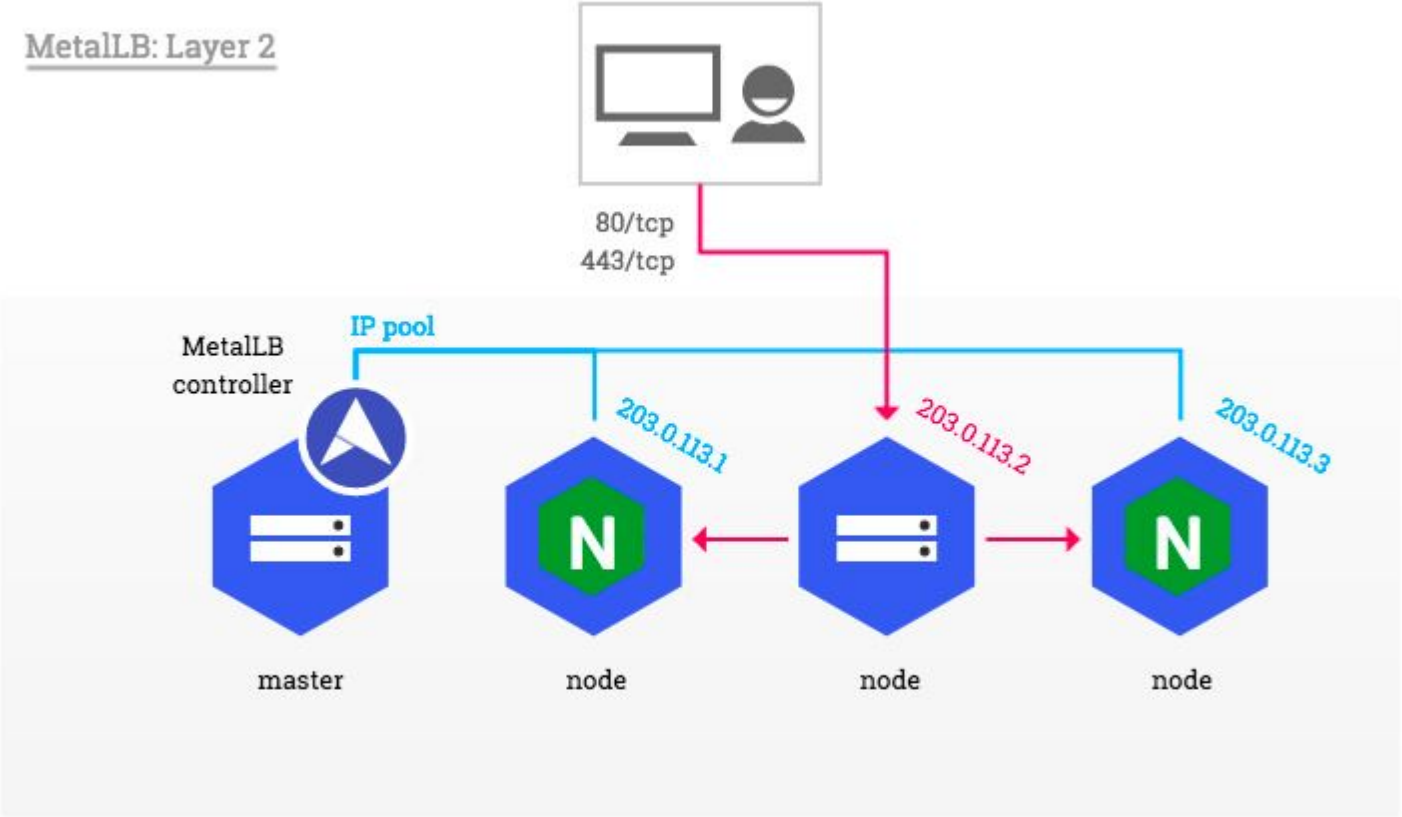
```yaml
apiVersion: v1
data:
  tls.crt: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURMVENDQWhXZ0F3SUJBZ0lK
  tls.key: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFb3dJQkFBS0NBUUVB
kind: Secret
metadata:
  name: webtest1secret
  namespace: default
type: Opaque
```

Secret: webtest2

```yaml
apiVersion: v1
data:
  tls.crt: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURMVENDQWhXZ0F3SU
  tls.key: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFcEFJQkFBS
kind: Secret
metadata:
  name: webtest2secret
  namespace: default
type: Opaque
```

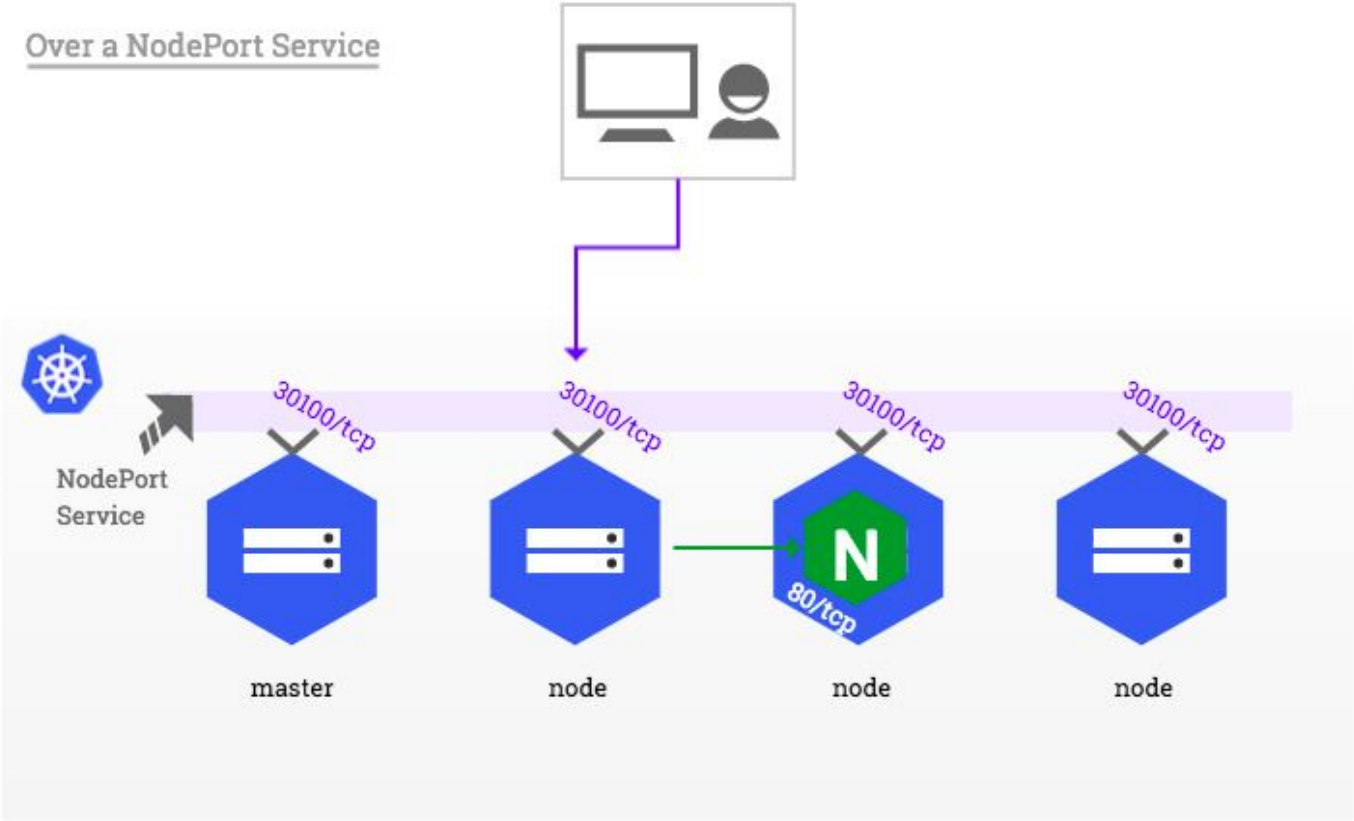
Ingress Network

- Ingress for On-prem strategy
  - MetalLB (L2 Load Balance)



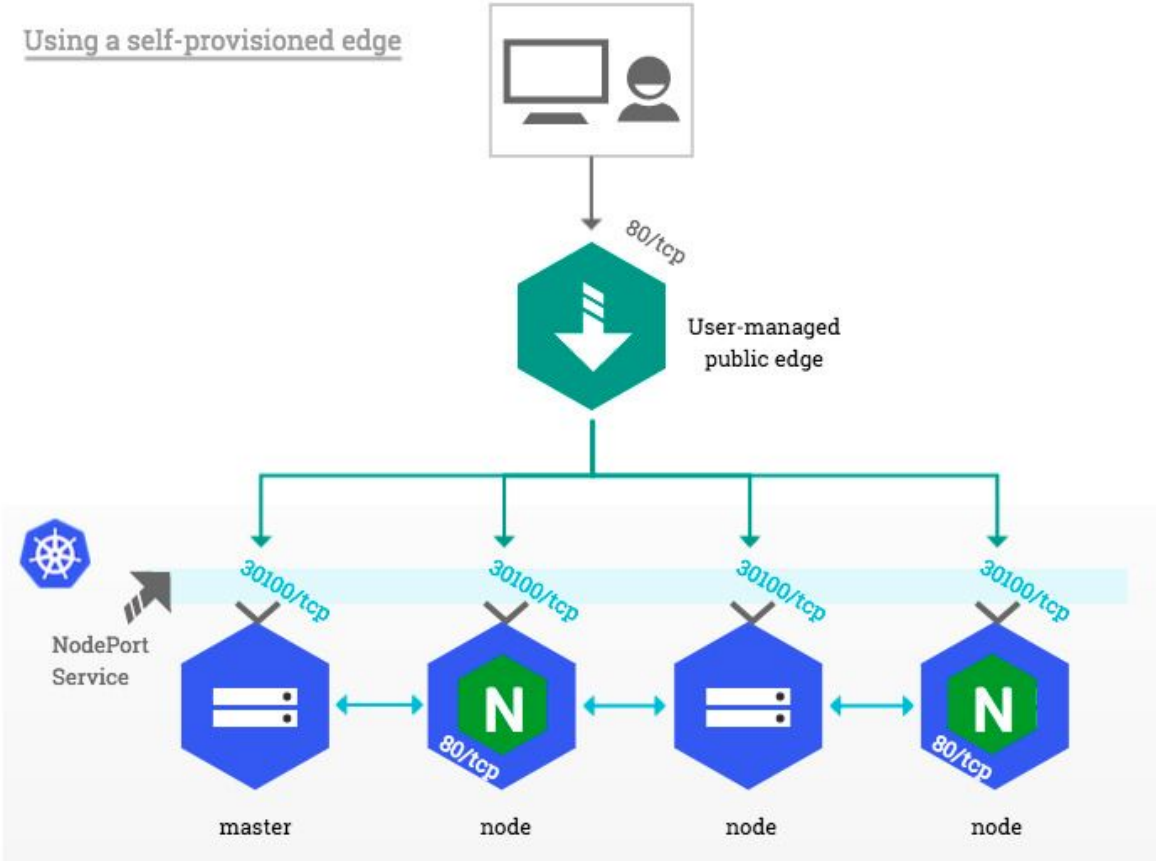
Ingress Network

- Ingress for On-prem strategy
  - NodePort (External Load Balance)



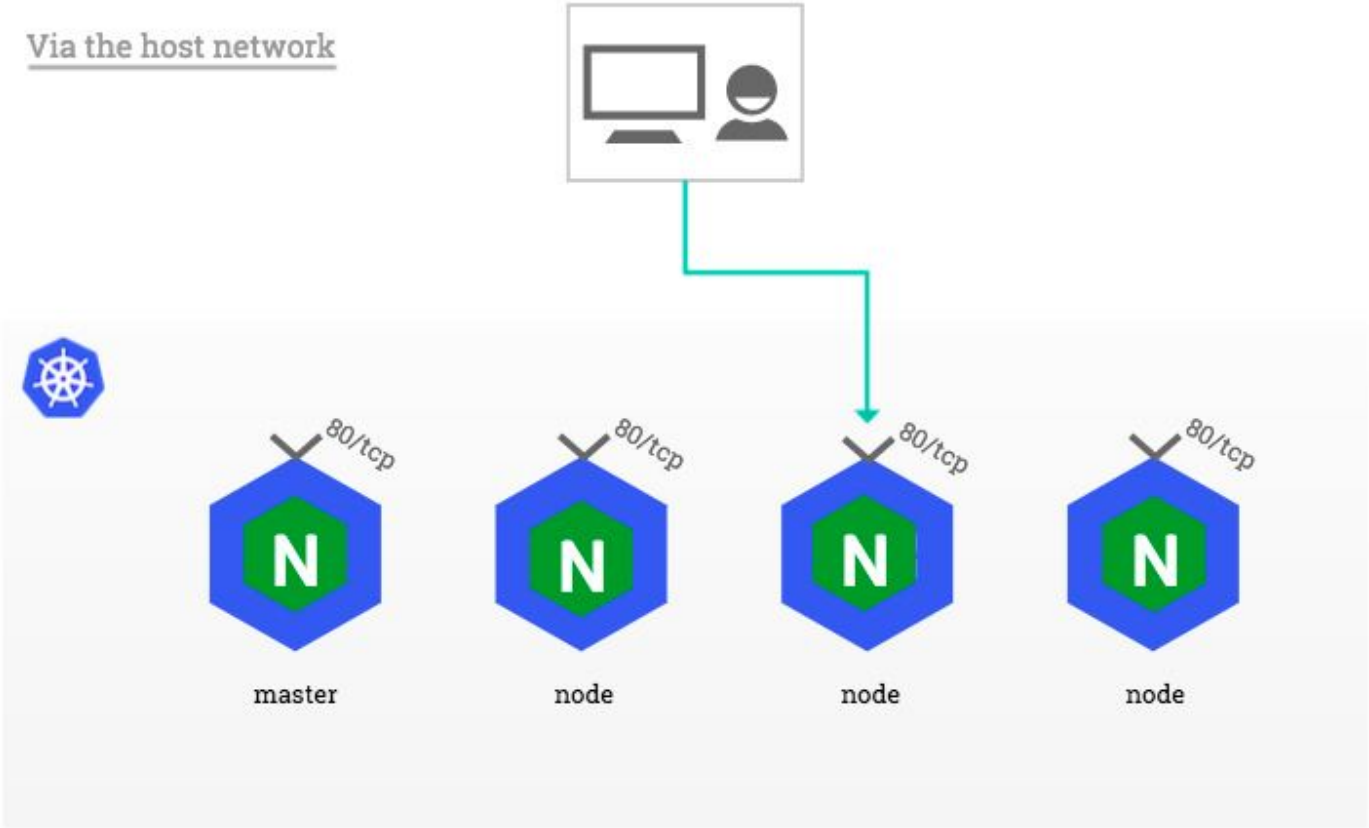
Ingress Network

- Ingress for On-prem strategy
  - NodePort (External Load Balance)



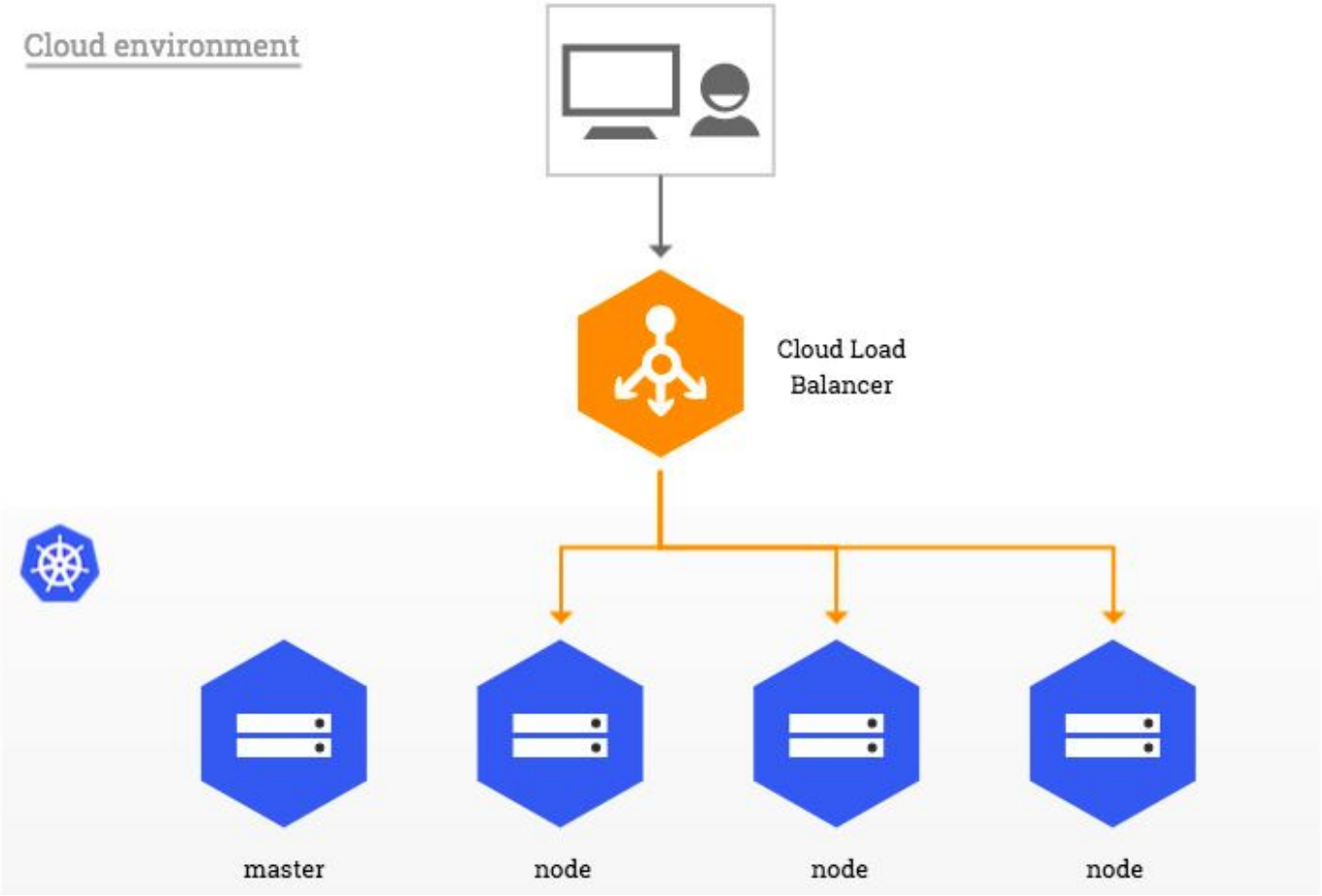
Ingress Network

- Ingress for On-prem strategy
  - Host network (Security concern, Not recommend)



Ingress Network

- Ingress for Cloud strategy



Ingress Network

- Ingress for Cloud strategy (Example: AWS)
  - Create Tag: kubernetes.io/cluster/<Cluster Name> for all resource
    - EC2 (VMWare Machine)
    - VPC
    - Subnet
    - Routing Table

Key	Value	
AZ	ap-southeast-1a	Show Column
Categories	traning	Show Column
Environment	uat	Show Column
Module	kubernetes farm	Show Column
Name	Training_DockerZerotoHero_StudentG5_1	Hide Column
Region	ap-southeast-1	Show Column
Zone	private	Show Column
kubernetes.io/cluster/TrainingAdvanceDockerwithK8SStudentG5	me	Show Column

AWS ¶

In AWS, we use a Network load balancer (NLB) to expose the Ingress-Nginx Controller behind a Service of `Type=LoadBalancer`.

> **Info**
>
> The provided templates illustrate the setup for legacy in-tree service load balancer for AWS NLB. AWS provides the documentation on how to use Network load balancing on Amazon EKS with AWS Load Balancer Controller.

kubernetes by Google

Ingress Network

AWS Load Balancer Controller v2.7 Search

Home
Welcome
How it works

Welcome

A Kubernetes controller for Elastic Load Balancers

contributions welcome | open issues 117 | status ga | license Apache-2.0

go report A+ | Watchers 113 | Stars 3.8k | Forks 1.4k | docker pulls 149M

AWS Load Balancer Controller

AWS Load Balancer Controller is a controller to help manage Elastic Load Balancers for a Kubernetes cluster.

- It satisfies Kubernetes Ingress resources by provisioning Application Load Balancers.
- It satisfies Kubernetes Service resources by provisioning Network Load Balancers.

Closed **Nginx Ingress controller not updating the target group for AWS EKS NLB** #9041
bhanugarg23 opened this issue on Sep 12, 2022 · 25 comments

sm-powell commented on Sep 20, 2022

AWS support have responded that they are aware of the issue, and if you want to continue to use AWS Load Balancer Controller with nginx ingress in Kubernetes version 1.23, you will need to wait for platform updates containing the pull request here: kubernetes/cloud-provider-aws#302

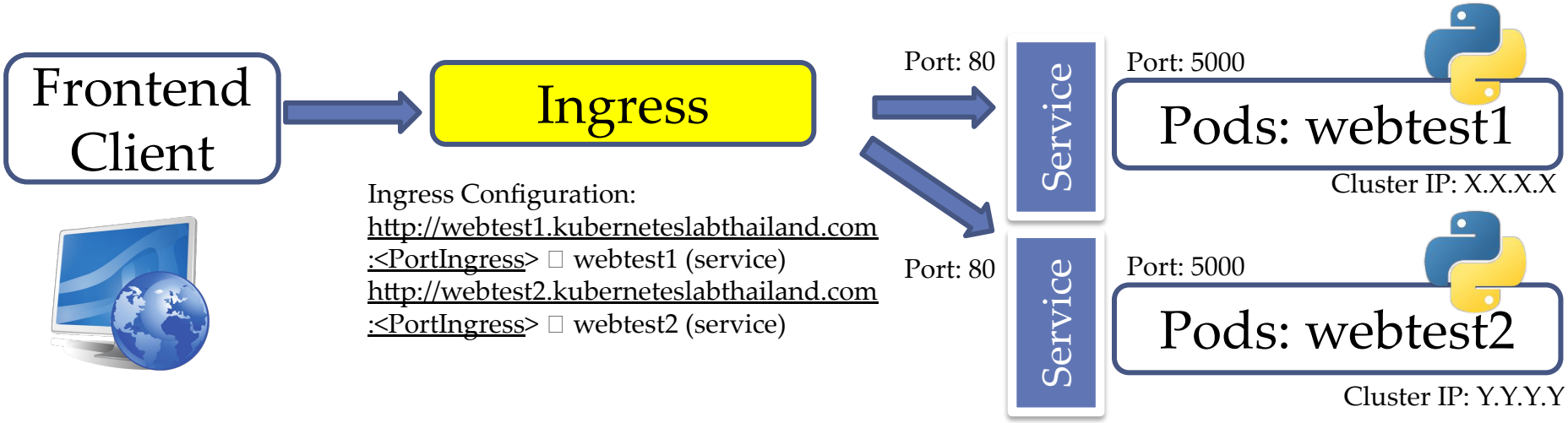
They also note that the underlying code supporting this has been moved to the legacy cloud provider, here: https://github.com/kubernetes/legacy-cloud-providers

I have not been able to investigate the implications of this yet, posting for everyone's information.

4

Ref: https://github.com/kubernetes-sigs/aws-load-balancer-controller

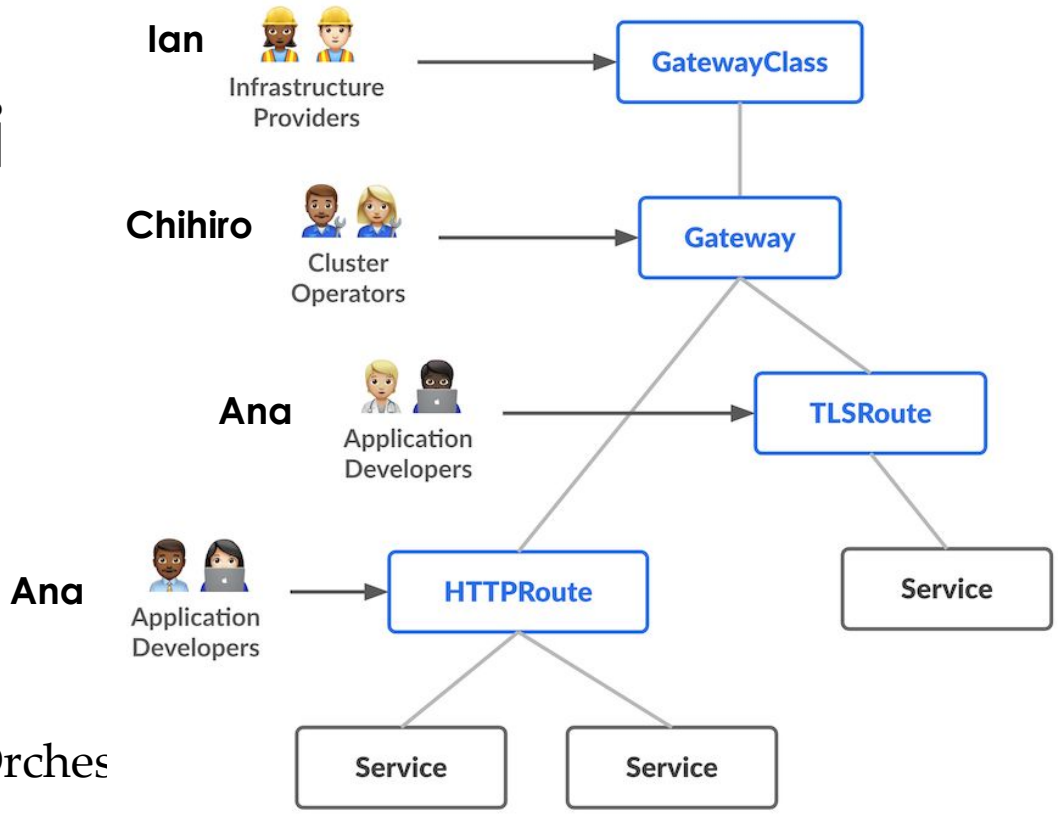
Workshop: Ingress Network



Gateway API

- Gateway API = Ingress + Load Balance + Service Mesh
- Gateway API is next generation of ingress
- Focus on network routing L4, L7 on Kubernetes network

gateway api



Gateway API

- What improve from ingress ?
  - Role-oriented: Persona
  - Portable: Can support multiple type of implement
  - Expressive: Support header matching, load balancer (Beyond from ingress annotation)
  - Extensible
    - Special Improvement:
      - GatewayClasses
      - Share Gateway and Cross-Namespace
      - Type Route (HTTP, Grpc) and Type Backend (Service, Function etc.)
      - Service mesh support: (Experimental)

Gateway API

- Difference between Gateway API and API Gateway?
  - An API Gateway is a general concept that describes anything that exposes capabilities of a backend service, while providing extra capabilities for traffic routing and manipulation, such as load balancing, request and response transformation, and sometimes more advanced features like authentication and authorization, rate limiting, and circuit breaking.

  - Gateway API is an interface, or set of resources, that model service networking in Kubernetes. One of the main resources is a Gateway, which declares the Gateway type (or class) to instantiate and its configuration. As a Gateway Provider, you can implement Gateway API to model Kubernetes service networking in an expressive, extensible, and role-oriented way.

Ref: https://gateway-api.sigs.k8s.io/

Gateway API

gateway api

Search

Overview
Introduction
Concepts
API Overview
Conformance
Roles and Personas
Security Model
Use Cases
Versioning
Service Mesh
Overview
GAMMA Initiative
Service Facets
Implementations
List
Comparison
FAQ
Glossary

Gateway Controller Implementation Status

- Acnodal EPIC
- Amazon Elastic Kubernetes Service (alpha)
- Apache APISIX (beta)
- Avi Kubernetes Operator (tech preview)
- Azure Application Gateway for Containers (GA)
- Cilium (beta)
- Contour (GA)
- Easegress (GA)
- Emissary-Ingress (Ambassador API Gateway) (alpha)
- Envoy Gateway (GA)
- Flomesh Service Mesh (beta)
- Gloo Gateway 2.0 (beta)
- Google Kubernetes Engine (GA)
- HAProxy Ingress (alpha)
- HAProxy Kubernetes Ingress Controller (GA)
- HashiCorp Consul
- Istio (beta)
- Kong (GA)
- Kuma (GA)
- LiteSpeed Ingress Controller
- NGINX Gateway Fabric (GA)
- ngrok (preview)
- STUNner (beta)
- Traefik (alpha)

gateway api

Search

Overview
Introduction
Concepts
API Overview
Conformance
Roles and Personas
Security Model
Use Cases
Versioning
Service Mesh
Overview
GAMMA Initiative
Service Facets
Implementations
List
Comparison
FAQ
Glossary

- Tyk (work in progress)
- WSO2 APK (GA)

Service Mesh Implementation Status

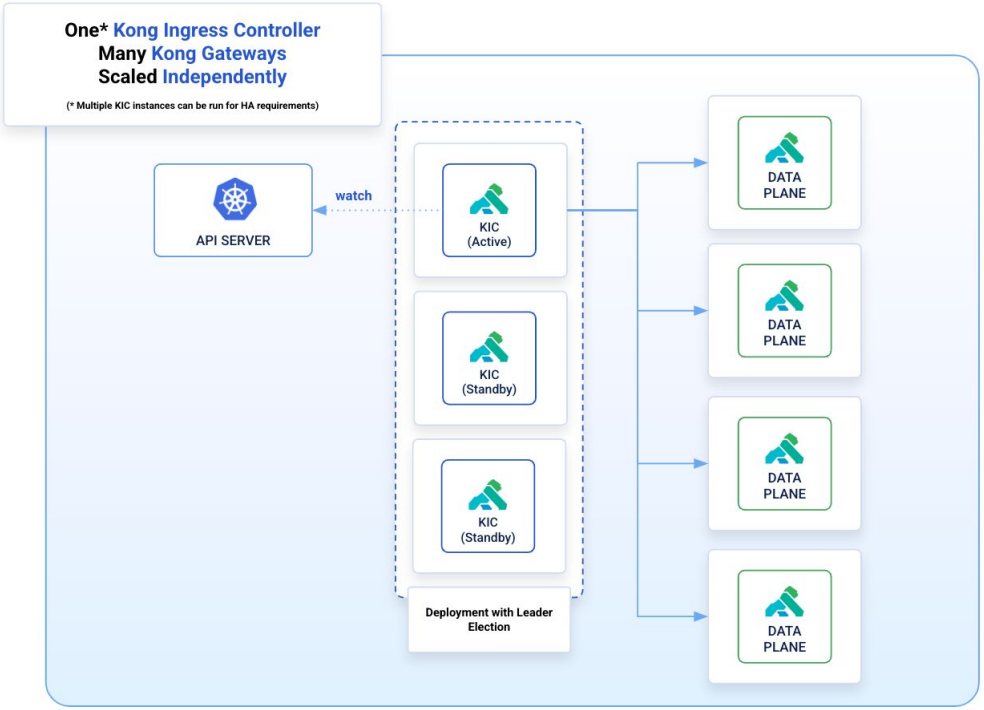
- Istio (experimental)
- Kuma (GA)
- Linkerd (experimental)

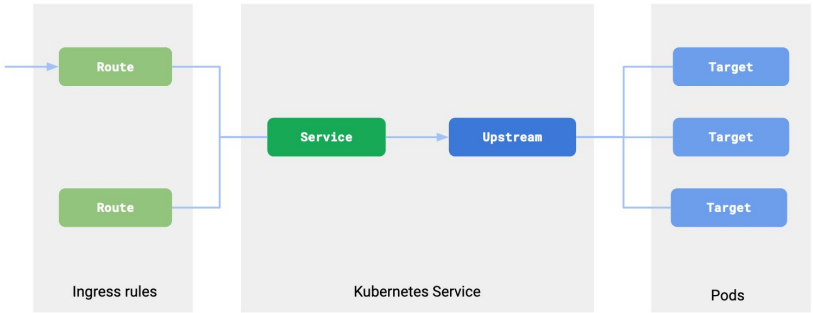
Integrations

- Flagger (public preview)
- cert-manager (alpha)
- argo-rollouts (alpha)
- Knative (alpha)
- Kuadrant (work in progress)

kubernetes by Google

Workshop: Ingress Network





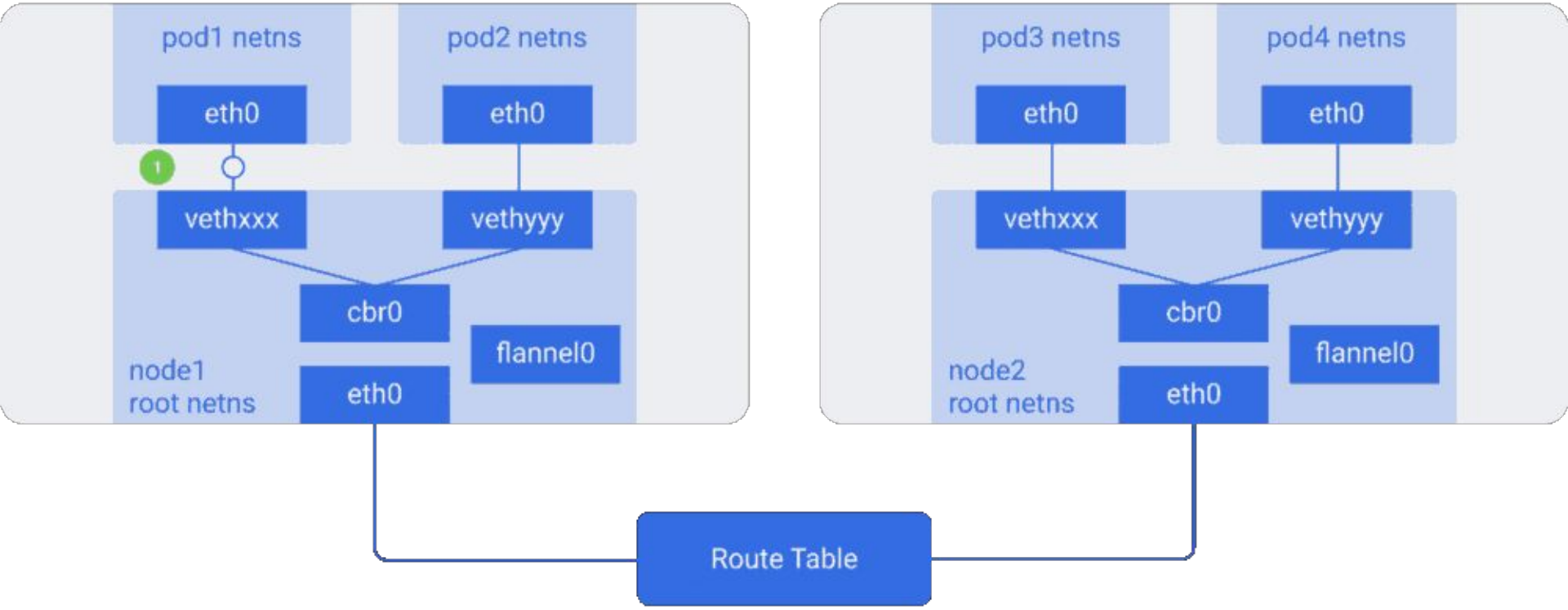
Security

Security

- Kubernetes have several security enhancement for manage in farm
  - **Network Policy**: Control policy for allow/deny connection between endpoint
  - **Volume Policy**: Control/Limit total volume attached to node
  - **Resource Usage Policy**: Control lower/upper acceptable range in namespace (LimitRange)
  - **Resource Consumption Policy**: Limit maximum consumption policy in namespace (Quota)
  - **Access Control Policy**: Deny/Allow fine-grained permission by RBAC, Rule etc
  - **Pod Security Admission (PSA)**: Control security enforcement context
  - **Encryption Provider:** Make secret great again

Security

- Network Policy:
  - Within same namespace
    - Each service will call by <service> or <service>.<namespace>
    - Across service across namespaces will call <service>.<namespace>



Security

- Network Policy:
  - By default, pods are non-isolated; they accept traffic from any source.
  - Network policy will apply target by “podSelector”, “
  - We can configure network policy base on label
    - Ingress: Control traffic incoming to pods
      - Template:
        - ingress
          - from: <podSelector>/<namespaceSelector>
    - Egress: Control traffic outcoming from pods
      - Template:
        - Egress
          - to: <podSelector>/<namespaceSelector>
  - Network Policy type
    - podSelector: Control policy of pods’s in/out same namespace (Base on label)
    - namespaceSelector: Control policy of pods’s in/out between namespace (Base on label)
    - ipBlock: Control policy by block ip address (Block ip range)

Security

kubernetes

Documentation | Kubernetes Blog

Search

Home
Getting started
Concepts
Tasks
Install Tools
Administer a Cluster
Administration with kubeadm
Migrating from dockershim
Certificates
Manage Memory, CPU, and API Resources
Install a Network Policy Provider
Access Clusters Using the Kubernetes API
Access Services Running on Clusters
Advertise Extended Resources for a Node
Autoscale the DNS Service in a Cluster
Change the default StorageClass
Change the Reclaim Policy of a PersistentVolume

Kubernetes Documentation / Tasks / Administer a Cluster / Declare Network Policy

Declare Network Policy

This document helps you get started using the Kubernetes NetworkPolicy API to declare network policies that govern how pods communicate with each other.

> **Caution:** This section links to third party projects that provide functionality required by Kubernetes. The Kubernetes project authors aren't responsible for these projects. This page follows CNCF website guidelines by listing projects alphabetically. To add a project to this list, read the content guide before submitting a change.

Before you begin

You need to have a Kubernetes cluster, and the kubectl command-line tool must be configured to communicate with your cluster. If you do not already have a cluster, you can create one by using minikube or you can use one of these Kubernetes playgrounds:

- Katacoda
- Play with Kubernetes

Your Kubernetes server must be at or later than version v1.8. To check the version, enter `kubectl version`.

Make sure you've configured a network provider with network policy support. There are a number of network providers that support NetworkPolicy, including:

- Calico
- Cilium
- Kube-router
- Romana
- Weave Net

kubernetes by Google

Security

- Network Policy:

WorkShop_2.2_Security > ! policy-default-allow-egress.yml

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all
spec:
  podSelector: {}
  egress:
  - {}
  policyTypes:
  - Egress
```

WorkShop_2.2_Security > ! policy-default-allow-ingress.yml

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all
spec:
  podSelector: {}
  ingress:
  - {}
  policyTypes:
  - Ingress
```

WorkShop_2.2_Security > ! policy-backend-policy.yml

```yaml
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
```

WorkShop_2.2_Security > ! policy-frontend-policy.yml

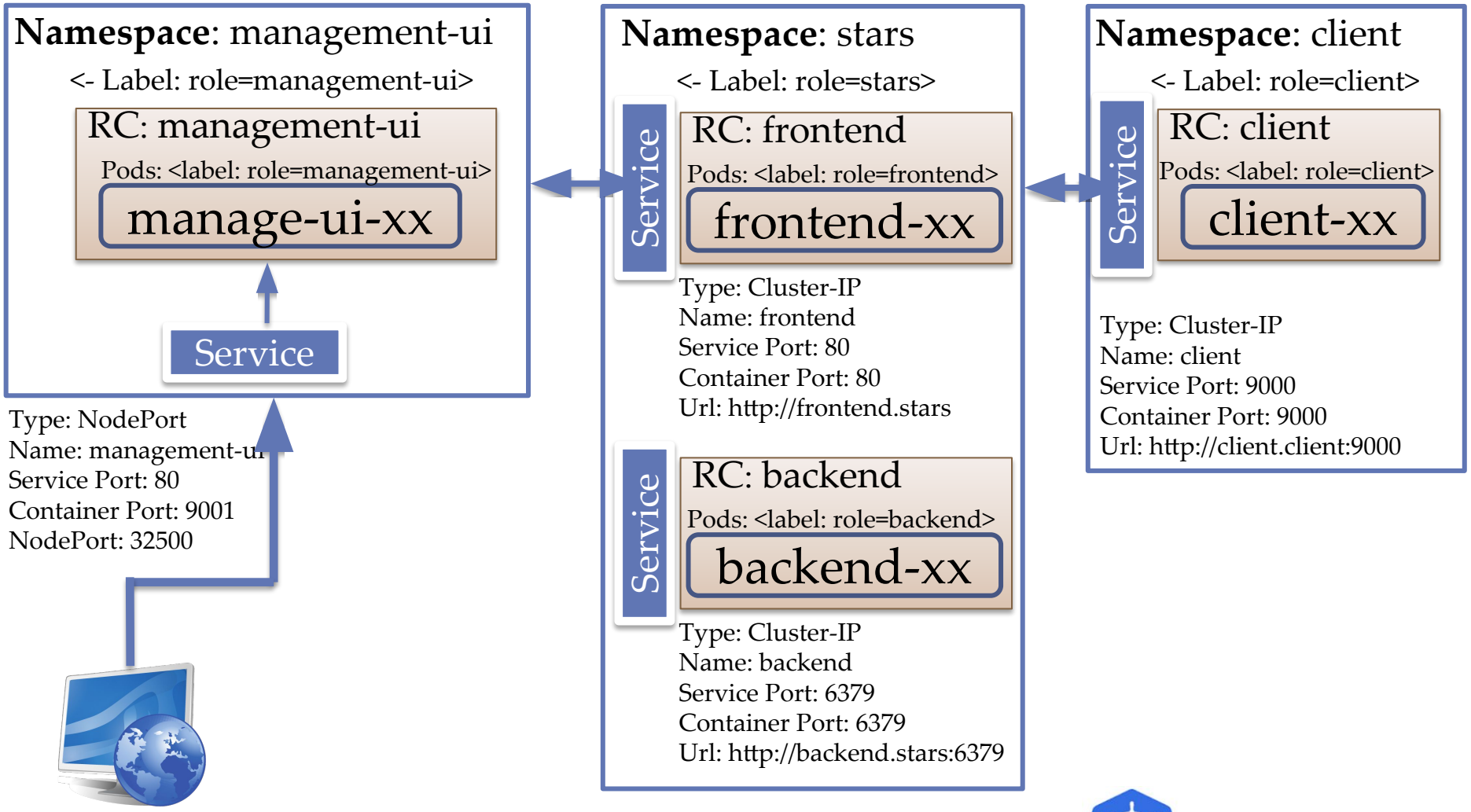
```yaml
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: TCP
          port: 80
```

WorkShop_2.2_Security > ! policy-template.yml

```yaml
kind: NetworkPolicy
metadata:
  name: policy-template
  namespace: stars
spec:
  podSelector:
    matchLabels:
      role: backend
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - ipBlock:
        cidr: 172.17.0.0/16
        except:
        - 172.17.1.0/24
    - namespaceSelector:
        matchLabels:
          role: management-ui
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 6379
  egress:
  - to:
    - ipBlock:
        cidr: 10.0.0.0/24
    ports:
    - protocol: TCP
      port: 5978
```

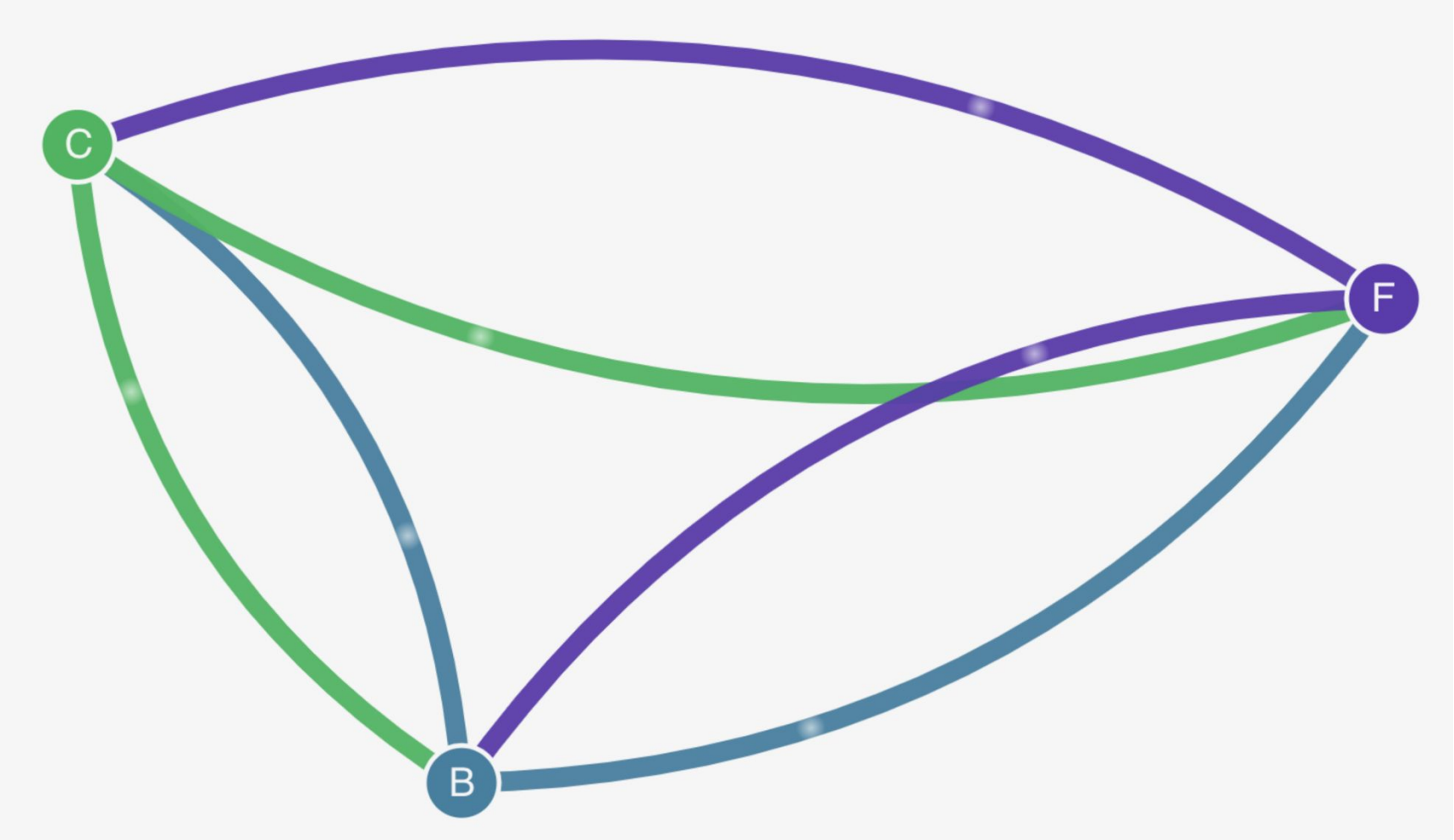
Security

- Network Policy:



Security

- Network Policy:



Security

- Network Policy:

Namespace: management-ui

<- Label: role=management-ui>

RC: management-ui

Pods: <label: role=management-ui>

manage-ui-xx

Service

Type: NodePort
Name: management-ui
Service Port: 80
Container Port: 9001
NodePort: 32500

Namespace: stars

<- Label: role=stars>

Service

RC: frontend

Pods: <label: role=frontend>

frontend-xx

Type: Cluster-IP
Name: frontend
Service Port: 80
Container Port: 80
Url: http://frontend.stars

X

Service

RC: backend

Pods: <label: role=backend>

backend-xx

Type: Cluster-IP
Name: backend
Service Port: 6379
Container Port: 6379
Url: http://backend.stars:6379

Namespace: client

<- Label: role=client>

Service

RC: client

Pods: <label: role=client>

client-xx

X

Type: Cluster-IP
Name: client
Service Port: 9000
Container Port: 9000
Url: http://client.client:9000

```
1  kind: NetworkPolicy
2  apiVersion: networking.k8s.io/v1
3  metadata:
4    name: default-deny
5    namespace: stars
6  spec:
7    podSelector:
8      matchLabels: {}
```

```
1  kind: NetworkPolicy
2  apiVersion: networking.k8s.io/v1
3  metadata:
4    name: default-deny
5    namespace: client
6  spec:
7    podSelector:
8      matchLabels: {}
```


kubernetes by Google

Security

- Network Policy:

Security

- Network Policy:

Namespace: management-ui

<- Label: role=management-ui>

RC: management-ui

Pods: <label: role=management-ui>

manage-ui-xx

Service

Type: NodePort
Name: management-ui
Service Port: 80
Container Port: 9001
NodePort: 32500

Namespace: stars

<- Label: role=stars>

Service

RC: frontend

Pods: <label: role=frontend>

frontend-xx

Type: Cluster-IP
Name: frontend
Service Port: 80
Container Port: 80
Url: http://frontend.stars

X

Y

Service

RC: backend

Pods: <label: role=backend>

backend-xx

Type: Cluster-IP
Name: backend
Service Port: 6379
Container Port: 6379
Url: http://backend.stars:6379

Namespace: client

<- Label: role=client>

Service

RC: client

Pods: <label: role=client>

client-xx

X

Y

Type: Cluster-IP
Name: client
Service Port: 9000
Container Port: 9000
Url: http://client.client:9000

```yaml
kind: NetworkPolicy
apiVersion: extensions/v1beta1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

```yaml
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Kubernetes: P[illegible]d Orchestration

kubernetes by Google

Security

- Network Policy:

Security

- Network Policy:

Namespace: management-ui

<- Label: role=management-ui>

RC: management-ui

Pods: <label: role=management-ui>

manage-ui-xx

Service

Type: NodePort
Name: management-ui
Service Port: 80
Container Port: 9001
NodePort: 32500

Namespace: stars

<- Label: role=stars>

Service

RC: frontend

Pods: <label: role=frontend>

frontend-xx

Y

X

Type: Cluster-IP
Name: frontend
Service Port: 80
Container Port: 80
Url: http://frontend.stars

Y

Y

Servi

RC: backend

Pods: <label: role=backend>

backend-xx

Type: Cluster-IP
Name: backend
Service Port: 6379
Container Port: 6379
Url: http://backend.stars:6379

Namespace: client

<- Label: role=client>

Service

RC: client

Pods: <label: role=client>

client-xx

X

Y

Type: Cluster-IP
Name: client
Service Port: 9000
Container Port: 9000
Url: http://client.client:9000

```yaml
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
```

```yaml
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: TCP
          port: 80
```

Kubernetes: Pr d Orchestration

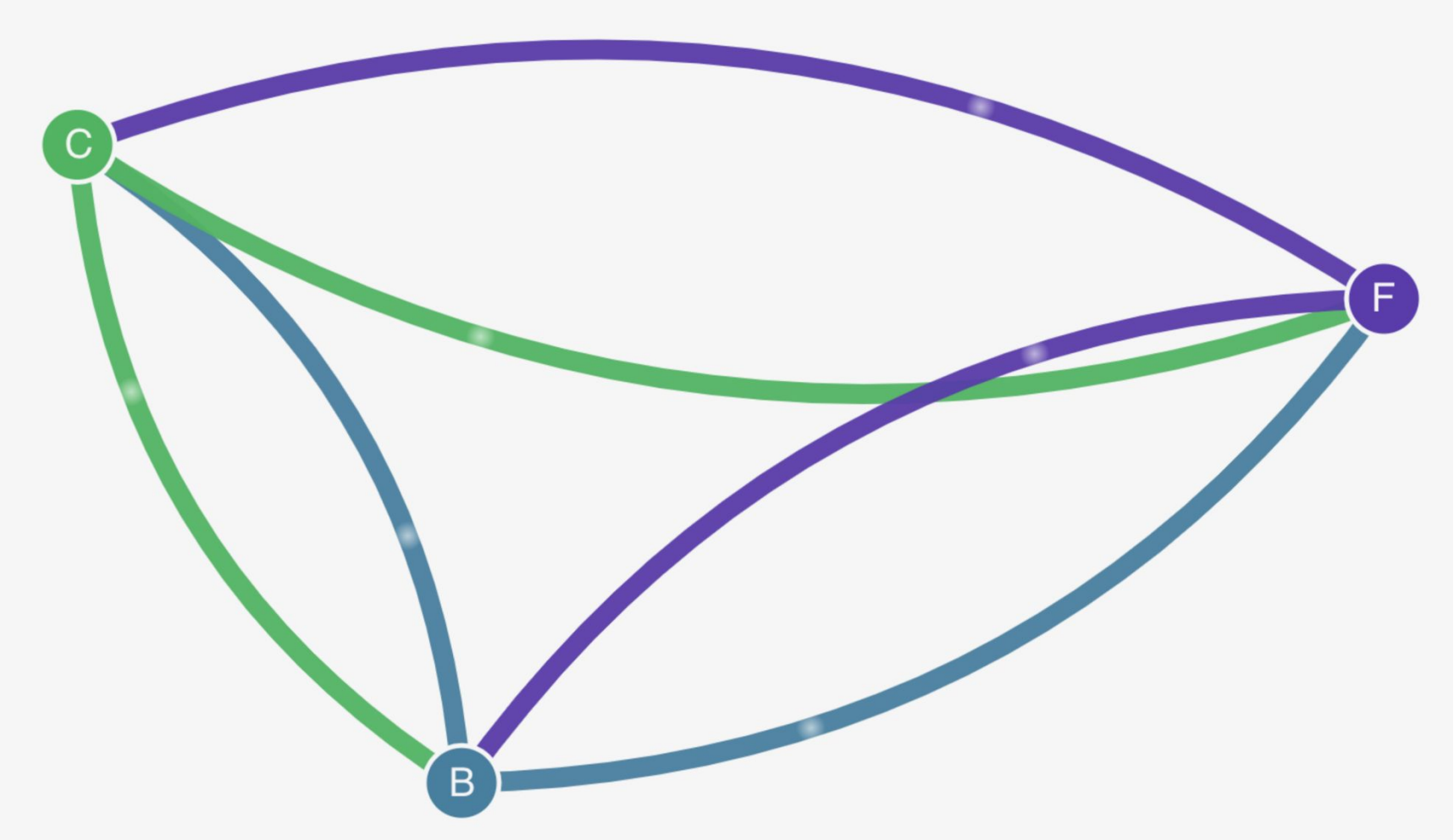
kubernetes by Google

Security

- Network Policy:

Workshop: Security

- Part: Network Policy



Security

- Volume Policy:
  - Kubernetes have default limit volume attached per host
  - Custom this value by environment variable: KUBE_MAX_PD_VOLS

Node-specific Volume Limits

This page describes the maximum number of volumes that can be attached to a Node for various cloud providers.

Cloud providers like Google, Amazon, and Microsoft typically have a limit on how many volumes can be attached to a Node. It is important for Kubernetes to respect those limits. Otherwise, Pods scheduled on a Node could get stuck waiting for volumes to attach.

- Kubernetes default limits
- Custom limits
- Dynamic volume limits

Kubernetes default limits

The Kubernetes scheduler has default limits on the number of volumes that can be attached to a Node:

Cloud service	Maximum volumes per Node
Amazon Elastic Block Store (EBS)	39
Google Persistent Disk	16
Microsoft Azure Disk Storage	16

Ref: https://kubernetes.io/docs/concepts/storage/storage-limits/

Security

- Resource Usage Policy:
  - Control by "ResourceQuota"

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: webtest-quota
  labels:
    name: webtest_quota
    owner: Praparn_L
    version: "1.0"
    module: Quota
    environment: development
spec:
  hard:
    pods: "4"
    requests.cpu: "1"
    requests.memory: 1Gi
    limits.cpu: "4"
    limits.memory: 4Gi
```

- Resource Consume Policy:
  - Control by "LimitRange"

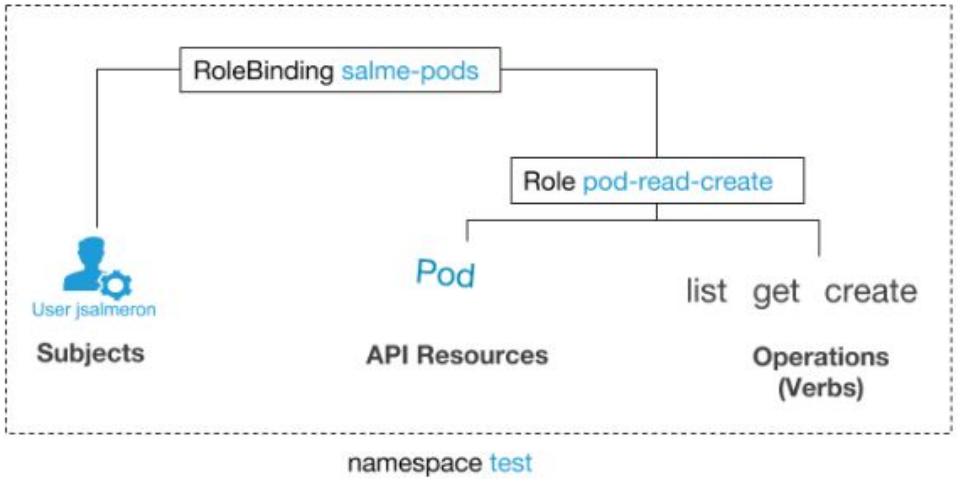
```yaml
apiVersion: v1
kind: LimitRange
metadata:
  name: webtest_limit
  labels:
    name: webtest_limit
    owner: Praparn_L
    version: "1.0"
    module: LimitRange
    environment: development
spec:
  limits:
  - max:
      cpu: "1"
      memory: 1Gi
    min:
      cpu: 200m
      memory: 6Mi
    type: Pod
  - default:
      cpu: 300m
      memory: 200Mi
    defaultRequest:
      cpu: 200m
      memory: 100Mi
    max:
      cpu: "1"
      memory: 1Gi
    min:
      cpu: 100m
      memory: 3Mi
    type: Container
```

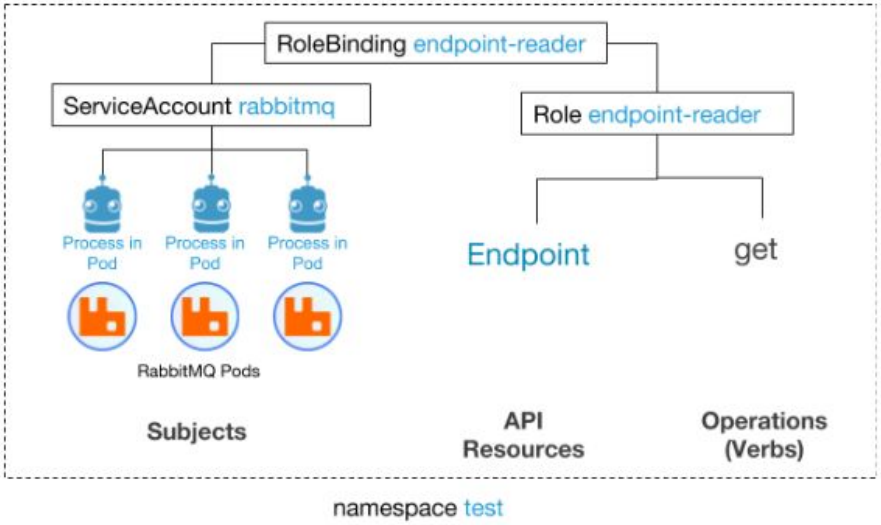
Security

- Access Control Policy
  - Normally Kubernetes manage privilege via RBAC (Role-base access control) for flexibility to change and assign privilege in role
  - Account for access with cluster will define to “User Account/Service Account"
  - RBAC normally describe in 4 categories
    - Role and ClusterRole:
    - RoleBinding and ClusterRoleBinding:
    - Referring to Resources:
    - Aggregated ClusterRoles:

Security

- Access Control Policy
  - **Account Management**
    - User accounts are for humans. Service accounts are for processes, which run in pods.
    - User accounts are global scope. Names must be unique across all namespaces of a cluster.
    - Auditing considerations for humans and service accounts may differ.





Ref: https://www.cncf.io/blog/2018/08/01/demystifying-rbac-in-kubernetes/

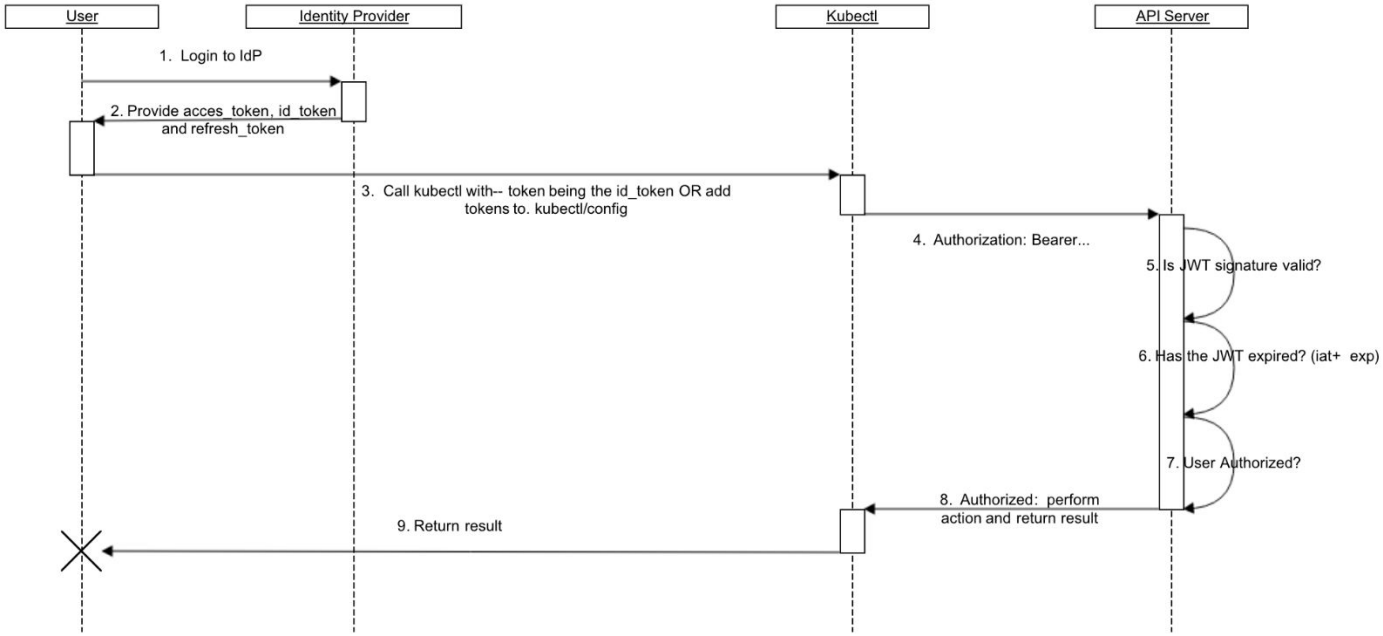
Security

- Access Control Policy
  - **Account Management**
    - Support OpenID.

OpenID Connect Tokens

OpenID Connect is a flavor of OAuth2 supported by some OAuth2 providers, notably Azure Active Directory, Salesforce, and Google. The protocol's main extension of OAuth2 is an additional field returned with the access token called an ID Token. This token is a JSON Web Token (JWT) with well known fields, such as a user's email, signed by the server.

To identify the user, the authenticator uses the `id_token` (not the `access_token`) from the OAuth2 token response as a bearer token. See above for how the token is included in a request.



Ref: https://kubernetes.io/docs/reference/access-authn-authz/authentication/

Security

- Access Control Policy
  - **Account Management**
    - Service Accounts

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: serviceaccount-readonly
```

    - User Accounts
      - No yaml file for create user account
      - Create via "openssl/kubectl" operate
        - private key for user and create csr (certificate sign request)(openssl)
        - generate final certificate from CA of Kubernetes CA (openssl)
        - create context for user via kubectl

Security

- Access Control Policy
  - **RBAC Operate**
    - Role and ClusterRole
    - Role: Describe rule for allow action on namespace level
    - ClusterRole: Describe rule for allow action on cluster level
    - Structure of syntax
      - rules
        - apiGroup[“”] □ Core API Group
        - resource[“”] □ King of resource to manage
        - verbs[“”] □ Action to allow

  - Default ClusterRole can check on this https://github.com/kubernetes/kubernetes/blob/master/plugin/pkg/auth/authorizer/rbac/bootstrappolicy/testdata/cluster-roles.yaml

Security

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: security
  name: templaterole
rules:
- apiGroups: [""] # "" indicates the core API group
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]
- apiGroups: ["batch"]
  resources: ["jobs"]
  verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]
- apiGroups: [""]
  resources: ["configmaps"]
  resourceNames: ["testconfigmap"]
  verbs: ["get"]
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["get", "list", "watch"]
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest1
  labels:
    name: webtest1
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  selector:
    matchLabels:
      name: webtest1
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: security
  name: rolereadonly
rules:
- apiGroups: ["", "apps"]
  resources: ["deployments", "replicasets", "pods"]
  verbs: ["get", "list", "watch"]
```

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

WorkShop_1.9_Job_CronJob > ! job.yml

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: linenotify
  labels:
    name: linenotify
    owner: Praparn_L
    version: "1.0"
    module: Job
    environment: development
spec:
  ttlSecondsAfterFinished: 30
  activeDeadlineSeconds: 60
  template:
    metadata:
      name: linenotify
    spec:
      containers:
      - name: linenotify
        image: labdocker/linenotify:onetime
```

Security

```yaml
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  labels:
  name: clusterrolereadonly
rules:
- apiGroups: ["", "apps", "extensions"]
  resources: ["*"]
  verbs: ["get", "list", "watch"]
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: templateclusterrole
  labels:
    rbac.kubernetesthailand.com/security: "true"
rules:
- apiGroups: [""] # "" indicates the core API group
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["get", "list"]
- apiGroups: ["batch"]
  resources: ["jobs"]
  verbs: ["get", "list"]
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["get", "list", "watch"]
```

Security

```
ubuntu@ip-10-0-1-78:~$ kubectl api-resources -o wide
NAME                              SHORTNAMES   APIGROUP                       NAMESPACED   KIND                          VERBS
bindings                                                                      true         Binding                       [create]
componentstatuses                 cs                                          false        ComponentStatus               [get list]
configmaps                        cm                                          true         ConfigMap                     [create delete deletecollection get list patch update watch]
endpoints                         ep                                          true         Endpoints                     [create delete deletecollection get list patch update watch]
events                            ev                                          true         Event                         [create delete deletecollection get list patch update watch]
limitranges                       limits                                      true         LimitRange                    [create delete deletecollection get list patch update watch]
namespaces                        ns                                          false        Namespace                     [create delete get list patch update watch]
nodes                             no                                          false        Node                          [create delete deletecollection get list patch update watch]
persistentvolumeclaims            pvc                                         true         PersistentVolumeClaim         [create delete deletecollection get list patch update watch]
persistentvolumes                 pv                                          false        PersistentVolume              [create delete deletecollection get list patch update watch]
pods                              po                                          true         Pod                           [create delete deletecollection get list patch update watch]
podtemplates                                                                  true         PodTemplate                   [create delete deletecollection get list patch update watch]
replicationcontrollers            rc                                          true         ReplicationController         [create delete deletecollection get list patch update watch]
resourcequotas                    quota                                       true         ResourceQuota                 [create delete deletecollection get list patch update watch]
secrets                                                                       true         Secret                        [create delete deletecollection get list patch update watch]
serviceaccounts                   sa                                          true         ServiceAccount                [create delete deletecollection get list patch update watch]
services                          svc                                         true         Service                       [create delete get list patch update watch]
mutatingwebhookconfigurations                  admissionregistration.k8s.io   false        MutatingWebhookConfiguration  [create delete deletecollection get list patch update watch]
validatingwebhookconfigurations                admissionregistration.k8s.io   false        ValidatingWebhookConfiguration [create delete deletecollection get list patch update watch]
customresourcedefinitions         crd,crds     apiextensions.k8s.io           false        CustomResourceDefinition      [create delete deletecollection get list patch update watch]
apiservices                                    apiregistration.k8s.io         false        APIService                    [create delete deletecollection get list patch update watch]
controllerrevisions                            apps                           true         ControllerRevision            [create delete deletecollection get list patch update watch]
daemonsets                        ds           apps                           true         DaemonSet                     [create delete deletecollection get list patch update watch]
deployments                       deploy       apps                           true         Deployment                    [create delete deletecollection get list patch update watch]
replicasets                       rs           apps                           true         ReplicaSet                    [create delete deletecollection get list patch update watch]
statefulsets                      sts          apps                           true         StatefulSet                   [create delete deletecollection get list patch update watch]
tokenreviews                                   authentication.k8s.io          false        TokenReview                   [create]
localsubjectaccessreviews                      authorization.k8s.io           true         LocalSubjectAccessReview      [create]
selfsubjectaccessreviews                       authorization.k8s.io           false        SelfSubjectAccessReview       [create]
selfsubjectrulesreviews                        authorization.k8s.io           false        SelfSubjectRulesReview        [create]
subjectaccessreviews                           authorization.k8s.io           false        SubjectAccessReview           [create]
horizontalpodautoscalers          hpa          autoscaling                    true         HorizontalPodAutoscaler       [create delete deletecollection get list patch update watch]
cronjobs                          cj           batch                          true         CronJob                       [create delete deletecollection get list patch update watch]
jobs                                           batch                          true         Job                           [create delete deletecollection get list patch update watch]
certificatesigningrequests        csr          certificates.k8s.io            false        CertificateSigningRequest     [create delete deletecollection get list patch update watch]
leases                                         coordination.k8s.io            true         Lease                         [create delete deletecollection get list patch update watch]
bgpconfigurations                              crd.projectcalico.org          false        BGPConfiguration              [delete deletecollection get list patch create update watch]
bgppeers                                       crd.projectcalico.org          false        BGPPeer                       [delete deletecollection get list patch create update watch]
blockaffinities                                crd.projectcalico.org          false        BlockAffinity                 [delete deletecollection get list patch create update watch]
clusterinformations                            crd.projectcalico.org          false        ClusterInformation            [delete deletecollection get list patch create update watch]
felixconfigurations                            crd.projectcalico.org          false        FelixConfiguration            [delete deletecollection get list patch create update watch]
globalnetworkpolicies                          crd.projectcalico.org          false        GlobalNetworkPolicy           [delete deletecollection get list patch create update watch]
globalnetworksets                              crd.projectcalico.org          false        GlobalNetworkSet              [delete deletecollection get list patch create update watch]
hostendpoints                                  crd.projectcalico.org          false        HostEndpoint                  [delete deletecollection get list patch create update watch]
ipamblocks                                     crd.projectcalico.org          false        IPAMBlock                     [delete deletecollection get list patch create update watch]
ipamconfigs                                    crd.projectcalico.org          false        IPAMConfig                    [delete deletecollection get list patch create update watch]
ipamhandles                                    crd.projectcalico.org          false        IPAMHandle                    [delete deletecollection get list patch create update watch]
ippools                                        crd.projectcalico.org          false        IPPool                        [delete deletecollection get list patch create update watch]
networkpolicies                                crd.projectcalico.org          true         NetworkPolicy                 [delete deletecollection get list patch create update watch]
networksets                                    crd.projectcalico.org          true         NetworkSet                    [delete deletecollection get list patch create update watch]
events                            ev           events.k8s.io                  true         Event                         [create delete deletecollection get list patch update watch]
ingresses                         ing          extensions                     true         Ingress                       [create delete deletecollection get list patch update watch]
ingresses                         ing          networking.k8s.io              true         Ingress                       [create delete deletecollection get list patch update watch]
networkpolicies                   netpol       networking.k8s.io              true         NetworkPolicy                 [create delete deletecollection get list patch update watch]
runtimeclasses                                 node.k8s.io                    false        RuntimeClass                  [create delete deletecollection get list patch update watch]
poddisruptionbudgets              pdb          policy                         true         PodDisruptionBudget           [create delete deletecollection get list patch update watch]
podsecuritypolicies               psp          policy                         false        PodSecurityPolicy             [create delete deletecollection get list patch update watch]
clusterrolebindings                            rbac.authorization.k8s.io      false        ClusterRoleBinding            [create delete deletecollection get list patch update watch]
clusterroles                                   rbac.authorization.k8s.io      false        ClusterRole                   [create delete deletecollection get list patch update watch]
rolebindings                                   rbac.authorization.k8s.io      true         RoleBinding                   [create delete deletecollection get list patch update watch]
roles                                          rbac.authorization.k8s.io      true         Role                          [create delete deletecollection get list patch update watch]
priorityclasses                   pc           scheduling.k8s.io              false        PriorityClass                 [create delete deletecollection get list patch update watch]
csidrivers                                     storage.k8s.io                 false        CSIDriver                     [create delete deletecollection get list patch update watch]
csinodes                                       storage.k8s.io                 false        CSINode                       [create delete deletecollection get list patch update watch]
storageclasses                    sc           storage.k8s.io                 false        StorageClass                  [create delete deletecollection get list patch update watch]
volumeattachments                              storage.k8s.io                 false        VolumeAttachment              [create delete deletecollection get list patch update watch]
ubuntu@ip-10-0-1-78:~$
```

Security

kubernetes.io/docs/reference/generated/kubernetes-api/v1.16/#-strong-api-overview-strong-

Apps | NMac Ked - Mac... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | Taiwan | NGINX | Kubernetes | PWA_Progressive_... | MYSQL_Cluster | GeneralKB | Nodejs | mongodb

Overview

WORKLOADS APIS

- Container v1 core
- CronJob v1beta1 batch
- DaemonSet v1 apps
- Deployment v1 apps
- Job v1 batch
- Pod v1 core
- ReplicaSet v1 apps
- ReplicationController v1 core
- StatefulSet v1 apps

SERVICE APIS

- Endpoints v1 core
- EndpointSlice v1alpha1 discovery.k8s.io
- Ingress v1beta1 networking.k8s.io
- Service v1 core

CONFIG AND STORAGE APIS

- ConfigMap v1 core
- CSIDriver v1beta1 storage.k8s.io
- CSINode v1beta1 storage.k8s.io
- Secret v1 core
- PersistentVolumeClaim v1 core
- StorageClass v1 storage.k8s.io
- Volume v1 core
- VolumeAttachment v1 storage.k8s.io

API OVERVIEW

Welcome to the Kubernetes API. You can use the Kubernetes API to read and write Kubernetes resource objects via a Kubernetes API endpoint.

Resource Categories

This is a high-level overview of the basic types of resources provide by the Kubernetes API and their primary functions.

Workloads are objects you use to manage and run your containers on the cluster.

Discovery & LB resources are objects you use to "stitch" your workloads together into an externally accessible, load-balanced Service.

Config & Storage resources are objects you use to inject initialization data into your applications, and to persist data that is external to your container.

Cluster resources objects define how the cluster itself is configured; these are typically used only by cluster operators.

Metadata resources are objects you use to configure the behavior of other resources within the cluster, such as `HorizontalPodAutoscaler` for scaling workloads.

Resource Objects

Resource objects typically have 3 components:

- **Resource ObjectMeta**: This is metadata about the resource, such as its name, type, api version, annotations, and labels. This contains fields that maybe updated both by the end user and the system (e.g. annotations).
- **ResourceSpec**: This is defined by the user and describes the desired state of system. Fill this in when creating or updating an object.
- **ResourceStatus**: This is filled in by the server and reports the current state of the system. In most cases, users don't need to change this.

Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.16/

kubernetes by Google

Security

- Access Control Policy
  - **RBAC Operate**
    - <u>RoleBinding and ClusterRoleBinding:</u>
    - Binding user accounts/system accounts to role or clusterrole

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: rolebindinglabsecurity
  namespace: security
subjects:
- kind: User
  name: labreadonly
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: rolereadonly
  apiGroup: rbac.authorization.k8s.io
```

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard

---

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kubernetes-dashboard
```

Security

- Access Control Policy
  - **RBAC Operate**
  - Referring to Resource
    - Identification "subresource" under main resource
    - Identification "specific resource" under main resource

WorkShop_2.2_Security > security-role-subresource.yml

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: security
  name: templaterolesubresource
rules:
- apiGroups: [""]
  resources: ["pods/log"]
  verbs: ["get"]
- apiGroups: [""]
  resources: ["configmaps"]
  resourceNames: ["testconfigmap"]
  verbs: ["update", "get"]
```

Security

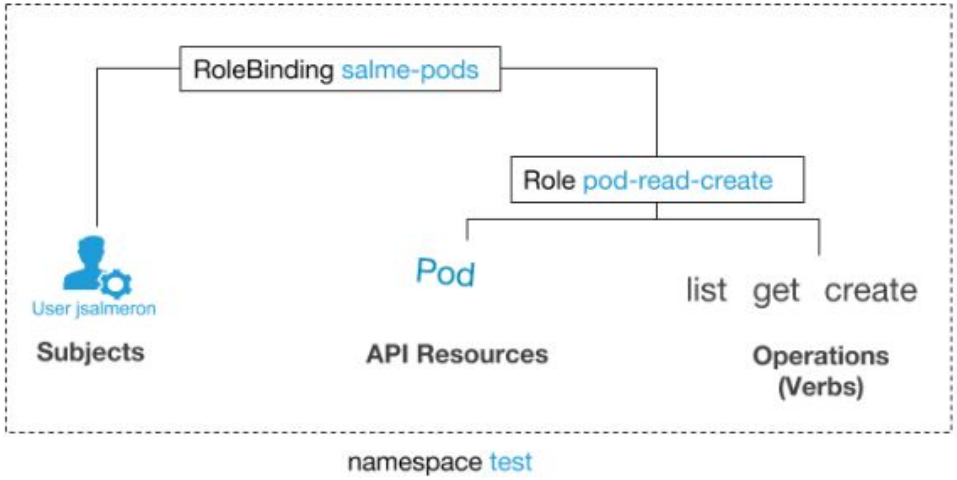
- Access Control Policy:
  - **RBAC Operate**
  - Aggregated ClusterRoles
  - Combine of multiple cluster roles
    - Match by label and rule will automatic filled

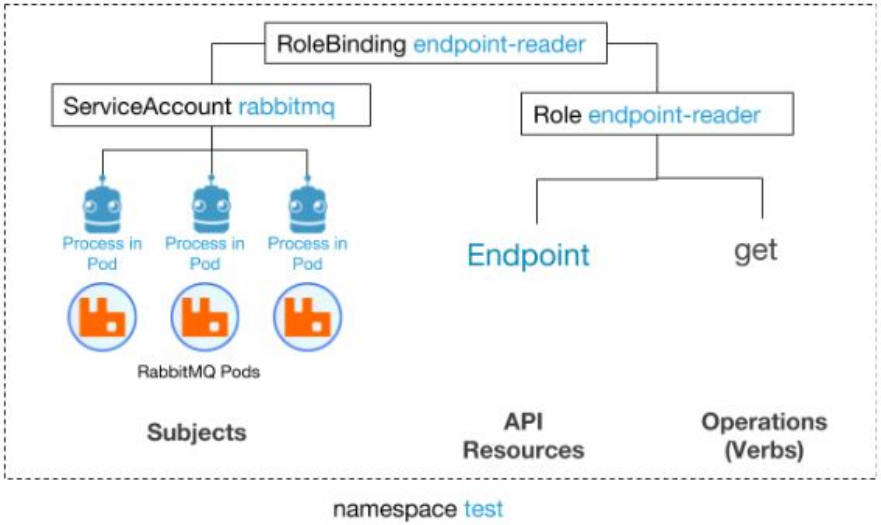
```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: templateaggregateclusterrole
  namespace: security
aggregationRule:
  clusterRoleSelectors:
  - matchLabels:
      rbac.kubernetesthailand.com/security: "true"
rules: [] # Rules are automatically filled in by the controller
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: templateclusterrole
  labels:
    rbac.kubernetesthailand.com/security: "true"
rules:
- apiGroups: [""] # "" indicates the core API group
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["get", "list"]
- apiGroups: ["batch"]
  resources: ["jobs"]
  verbs: ["get", "list"]
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["get", "list", "watch"]
```

Workshop: Security

- Part: Access Control Policy






```
ubuntu@ip-10-0-1-78:~$ kubectl --context=labreadonly-context get pods
NAME      READY   STATUS    RESTARTS   AGE
webtest   1/1     Running   0          56m
ubuntu@ip-10-0-1-78:~$ kubectl --context=labreadonly-context delete pods/webtest
Error from server (Forbidden): pods "webtest" is forbidden: User "labreadonly" cannot delete resource "pods" in API group "" in the namespace "security"
ubuntu@ip-10-0-1-78:~$ kubectl --context=labreadonly-context run webtest --image=labdocker/nginx:http2 --port=443
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a future version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
Error from server (Forbidden): deployments.apps is forbidden: User "labreadonly" cannot create resource "deployments" in API group "apps" in the namespace "security"
ubuntu@ip-10-0-1-78:~$ kubectl --context=labreadonly-context run webtest --image=labdocker/nginx:http2 --port=443 -n=security
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a future version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
Error from server (Forbidden): deployments.apps is forbidden: User "labreadonly" cannot create resource "deployments" in API group "apps" in the namespace "security"
```

Security

- Pod Security Admission (Beta: 1.23):

PodSecurity graduates to Beta

PodSecurity moves to Beta. `PodSecurity` replaces the deprecated `PodSecurityPolicy` admission controller. `PodSecurity` is an admission controller that enforces Pod Security Standards on Pods in a Namespace based on specific namespace labels that set the enforcement level. In 1.23, the `PodSecurity` feature gate is enabled by default.

Security

- Pod Security Admission:
  - Next generation of Pod Security Policy (Obsolete PSP on 1.25)
  - Enforcement security policy both in level of "cluster/namespace/pods" with 3 mode for operate
    - **Enforcing**: Policy violations will cause the pod to be rejected.
    - **Audit**: Policy violations will trigger the addition of an audit annotation to the event recorded in the audit log, but are otherwise allowed
    - **Warn**: Policy violations will trigger a user-facing warning, but are otherwise allowed.
  - For each mode for operate. We can define rule with 3 profiles for apply
    - **Privilege**: Unrestricted policy (Privilege escalate)
    - **Baseline**: Minimally restrictive policy which prevents known privilege escalations
    - **Restrict**: Heavily restricted policy, following current Pod hardening best practices.

Security

- Privilege Profile:
  - Unrestrict at all !!!
  - Aim for infrastructure/system namespace
  - Manage by trust users

Security

- Baseline Profile:
  - Adoption for common containerized workload
  - Prevent know privilege escalate
  - Target for general application deployment
  - "Non-critical" system
  - (New Feature) Allow windows privilege for windows pods
    - NT AUTHORITY\SYSTEM
    - NT AUTHORITY\Local service
    - NT AUTHORITY\NetworkService

Ref: https://kubernetes.io/docs/concepts/security/pod-security-standards/

Security

- Baseline Profile:

Control	Policy
HostProcess	Windows pods offer the ability to run HostProcess containers which enables privileged access to the Windows node. Privileged access to the host is disallowed in the baseline policy. HostProcess pods are an **alpha** feature as of Kubernetes **v1.22**. **Restricted Fields** • `spec.securityContext.windowsOptions.hostProcess` • `spec.containers[*].securityContext.windowsOptions.hostProcess` • `spec.initContainers[*].securityContext.windowsOptions.hostProcess` • `spec.ephemeralContainers[*].securityContext.windowsOptions.hostProcess` **Allowed Values** • Undefined/nil • `false`
Host Namespaces	Sharing the host namespaces must be disallowed. **Restricted Fields** • `spec.hostNetwork` • `spec.hostPID` • `spec.hostIPC` **Allowed Values** • Undefined/nil • `false`
Privileged Containers	Privileged Pods disable most security mechanisms and must be disallowed. **Restricted Fields** • `spec.containers[*].securityContext.privileged` • `spec.initContainers[*].securityContext.privileged` • `spec.ephemeralContainers[*].securityContext.privileged` **Allowed Values** • Undefined/nil • `false`

Security

- Baseline Profile:

Capabilities	Adding additional capabilities beyond those listed below must be disallowed. **Restricted Fields** • `spec.containers[*].securityContext.capabilities.add` • `spec.initContainers[*].securityContext.capabilities.add` • `spec.ephemeralContainers[*].securityContext.capabilities.add` **Allowed Values** • Undefined/nil • `AUDIT_WRITE` • `CHOWN` • `DAC_OVERRIDE` • `FOWNER` • `FSETID` • `KILL` • `MKNOD` • `NET_BIND_SERVICE` • `SETFCAP` • `SETGID` • `SETPCAP` • `SETUID` • `SYS_CHROOT`
HostPath Volumes	HostPath volumes must be forbidden. **Restricted Fields** • `spec.volumes[*].hostPath` **Allowed Values** • Undefined/nil

Security

- Baseline Profile:

Host Ports	HostPorts should be disallowed, or at minimum restricted to a known list. **Restricted Fields** • `spec.containers[*].ports[*].hostPort` • `spec.initContainers[*].ports[*].hostPort` • `spec.ephemeralContainers[*].ports[*].hostPort` **Allowed Values** • Undefined/nil • Known list • `0`
AppArmor	On supported hosts, the `runtime/default` AppArmor profile is applied by default. The baseline policy should prevent overriding or disabling the default AppArmor profile, or restrict overrides to an allowed set of profiles. **Restricted Fields** • `metadata.annotations["container.apparmor.security.beta.kubernetes.io/*"]` **Allowed Values** • Undefined/nil • `runtime/default` • `localhost/*`

Security

- Baseline Profile:

SELinux	Setting the SELinux type is restricted, and setting a custom SELinux user or role option is forbidden. **Restricted Fields** • `spec.securityContext.seLinuxOptions.type` • `spec.containers[*].securityContext.seLinuxOptions.type` • `spec.initContainers[*].securityContext.seLinuxOptions.type` • `spec.ephemeralContainers[*].securityContext.seLinuxOptions.type` **Allowed Values** • Undefined/"" • `container_t` • `container_init_t` • `container_kvm_t` **Restricted Fields** • `spec.securityContext.seLinuxOptions.user` • `spec.containers[*].securityContext.seLinuxOptions.user` • `spec.initContainers[*].securityContext.seLinuxOptions.user` • `spec.ephemeralContainers[*].securityContext.seLinuxOptions.user` • `spec.securityContext.seLinuxOptions.role` • `spec.containers[*].securityContext.seLinuxOptions.role` • `spec.initContainers[*].securityContext.seLinuxOptions.role` • `spec.ephemeralContainers[*].securityContext.seLinuxOptions.role` **Allowed Values** • Undefined/""
`/proc` Mount Type	The default `/proc` masks are set up to reduce attack surface, and should be required. **Restricted Fields** • `spec.containers[*].securityContext.procMount` • `spec.initContainers[*].securityContext.procMount` • `spec.ephemeralContainers[*].securityContext.procMount` **Allowed Values** • Undefined/nil • `Default`

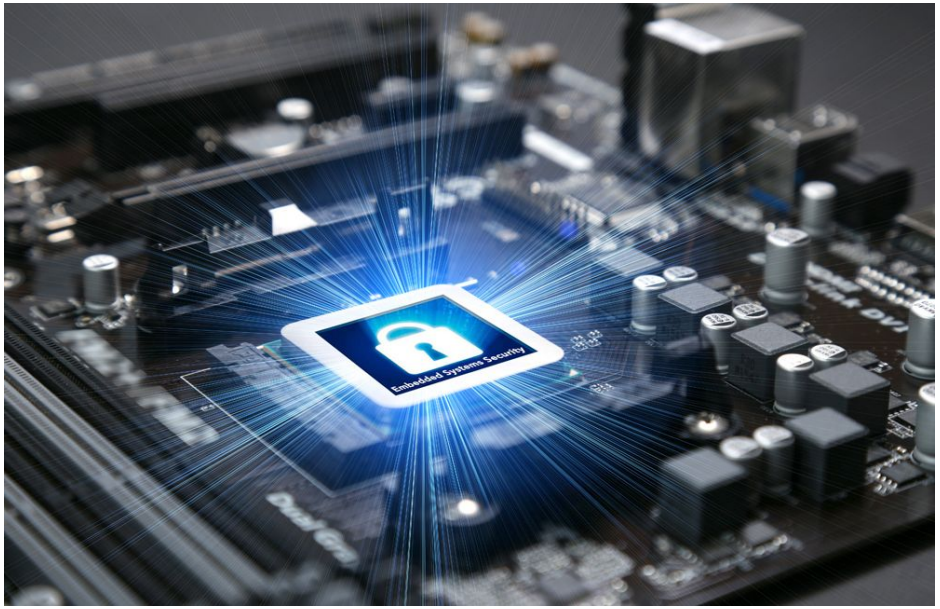
Security

- Baseline Profile:

Seccomp	Seccomp profile must not be explicitly set to `Unconfined` . **Restricted Fields** • `spec.securityContext.seccompProfile.type` • `spec.containers[*].securityContext.seccompProfile.type` • `spec.initContainers[*].securityContext.seccompProfile.type` • `spec.ephemeralContainers[*].securityContext.seccompProfile.type` **Allowed Values** • Undefined/nil • `RuntimeDefault` • `Localhost`
Sysctls	Sysctls can disable security mechanisms or affect all containers on a host, and should be disallowed except for an allowed "safe" subset. A sysctl is considered safe if it is namespaced in the container or the Pod, and it is isolated from other Pods or processes on the same Node. **Restricted Fields** • `spec.securityContext.sysctls[*].name` **Allowed Values** • Undefined/nil • `kernel.shm_rmid_forced` • `net.ipv4.ip_local_port_range` • `net.ipv4.ip_unprivileged_port_start` • `net.ipv4.tcp_syncookies` • `net.ipv4.ping_group_range`

Security

- Restricted Profile:
  - Hardest practice for security profile
  - Low trust user environment
  - "Critical" system
  - Base on "baseline" profile with enhancement more



Ref: https://kubernetes.io/docs/concepts/security/pod-security-standards/

Security

- Restricted Profile:

Volume Types	In addition to restricting HostPath volumes, the restricted policy limits usage of non-core volume types to those defined through PersistentVolumes. **Restricted Fields** • `spec.volumes[*].hostPath` • `spec.volumes[*].gcePersistentDisk` • `spec.volumes[*].awsElasticBlockStore` • `spec.volumes[*].gitRepo` • `spec.volumes[*].nfs` • `spec.volumes[*].iscsi` • `spec.volumes[*].glusterfs` • `spec.volumes[*].rbd` • `spec.volumes[*].flexVolume` • `spec.volumes[*].cinder` • `spec.volumes[*].cephfs` • `spec.volumes[*].flocker` • `spec.volumes[*].fc` • `spec.volumes[*].azureFile` • `spec.volumes[*].vsphereVolume` • `spec.volumes[*].quobyte` • `spec.volumes[*].azureDisk` • `spec.volumes[*].portworxVolume` • `spec.volumes[*].scaleIO` • `spec.volumes[*].storageos` • `spec.volumes[*].photonPersistentDisk` **Allowed Values** • Undefined/nil
Privilege Escalation (v1.8+)	Privilege escalation (such as via set-user-ID or set-group-ID file mode) should not be allowed. **Restricted Fields** • `spec.containers[*].securityContext.allowPrivilegeEscalation` • `spec.initContainers[*].securityContext.allowPrivilegeEscalation` • `spec.ephemeralContainers[*].securityContext.allowPrivilegeEscalation` **Allowed Values** • `false`

Security

- Restricted Profile:

Running as Non-root	Containers must be required to run as non-root users. **Restricted Fields** • `spec.securityContext.runAsNonRoot` • `spec.containers[*].securityContext.runAsNonRoot` • `spec.initContainers[*].securityContext.runAsNonRoot` • `spec.ephemeralContainers[*].securityContext.runAsNonRoot` **Allowed Values** • `true` The container fields may be undefined/ `nil` if the pod-level `spec.securityContext.runAsNonRoot` is set to `true` .
Non-root groups *(optional)*	Containers should be forbidden from running with a root primary or supplementary GID. **Restricted Fields** • `spec.securityContext.runAsGroup` • `spec.securityContext.supplementalGroups[*]` • `spec.securityContext.fsGroup` • `spec.containers[*].securityContext.runAsGroup` • `spec.initContainers[*].securityContext.runAsGroup` • `spec.ephemeralContainers[*].securityContext.runAsGroup` **Allowed Values** • Undefined/nil (except for `*.runAsGroup`) • Non-zero
Seccomp (v1.19+)	Seccomp profile must be explicitly set to one of the allowed values. Both the `Unconfined` profile and the *absence* of a profile are prohibited. **Restricted Fields** • `spec.securityContext.seccompProfile.type` • `spec.containers[*].securityContext.seccompProfile.type` • `spec.initContainers[*].securityContext.seccompProfile.type` • `spec.ephemeralContainers[*].securityContext.seccompProfile.type` **Allowed Values** • `RuntimeDefault` • `Localhost` The container fields may be undefined/ `nil` if the pod-level `spec.securityContext.seccompProfile.type` field is set appropriately. Conversely, the pod-level field may be undefined/ `nil` if _all_ container- level fields are set.

Security

- Restricted Profile:

Capabilities (v1.22+)	Containers must drop `ALL` capabilities, and are only permitted to add back the `NET_BIND_SERVICE` capability. **Restricted Fields** • `spec.containers[*].securityContext.capabilities.drop` • `spec.initContainers[*].securityContext.capabilities.drop` • `spec.ephemeralContainers[*].securityContext.capabilities.drop` **Allowed Values** • Any list of capabilities that includes `ALL` **Restricted Fields** • `spec.containers[*].securityContext.capabilities.add` • `spec.initContainers[*].securityContext.capabilities.add` • `spec.ephemeralContainers[*].securityContext.capabilities.add` **Allowed Values** • Undefined/nil • `NET_BIND_SERVICE`

Security

- Pods security standard for cluster/namespace
  - Customize for security profile on each mode
    - enforce: (baseline/restricted/privileged)
    - audit: (baseline/restricted/privileged)
    - warning: (baseline/restricted/privileged)
  - Enforce PSA for new namespace

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: namespace-psa
  labels:
    name: namespace-psa
    owner: Praparn_L
    version: "1.0"
    module: NameSpace
    environment: development
    pod-security.kubernetes.io/enforce: baseline
    pod-security.kubernetes.io/enforce-version: v1.29

    # We are setting these to our _desired_ `enforce` level.
    pod-security.kubernetes.io/audit: restricted
    pod-security.kubernetes.io/audit-version: v1.29
    pod-security.kubernetes.io/warn: restricted
    pod-security.kubernetes.io/warn-version: v1.29
```

Security

- Namespace Level: Enforce PSA existing namespace

```
ubuntu@ip-10-21-2-23:~$ kubectl get nodes
NAME          STATUS   ROLES                      AGE   VERSION
10.21.2.116   Ready    controlplane,etcd,worker   41d   v1.20.9
10.21.2.188   Ready    controlplane,etcd,worker   41d   v1.20.9
10.21.2.203   Ready    controlplane,etcd,worker   41d   v1.20.9
ubuntu@ip-10-21-2-23:~$ kubectl label --overwrite ns namespace-psa \
>   pod-security.kubernetes.io/enforce=restricted \
>   pod-security.kubernetes.io/enforce-version=v1.22 \
>   pod-security.kubernetes.io/audit: restricted \
>   pod-security.kubernetes.io/audit-version: v1.22 \
>   pod-security.kubernetes.io/warn: restricted \
>   pod-security.kubernetes.io/warn-version: v1.22
```

Security

- Cluster Level: Enforce PSA all namespace

```
ubuntu@ip-10-21-2-23:~$ kubectl get nodes
NAME          STATUS   ROLES                      AGE   VERSION
10.21.2.116   Ready    controlplane,etcd,worker   41d   v1.20.9
10.21.2.188   Ready    controlplane,etcd,worker   41d   v1.20.9
10.21.2.203   Ready    controlplane,etcd,worker   41d   v1.20.9
ubuntu@ip-10-21-2-23:~$   kubectl label --overwrite ns --all \
>   pod-security.kubernetes.io/enforce=restricted \
>   pod-security.kubernetes.io/enforce-version=v1.22 \
>   pod-security.kubernetes.io/audit: restricted \
>   pod-security.kubernetes.io/audit-version: v1.22 \
>   pod-security.kubernetes.io/warn: restricted \
>   pod-security.kubernetes.io/warn-version: v1.22
```

Security

- Pods security standard for pods/container
  - Permission to access object (Base on user/group id)
  - Security-Enhanced Linux (SELinux)
  - Privilege/Non-Privileged running
  - Limit some capability of root on pods/container (Linux capability)
  - Apparmor/Seccomp
  - AllowPrivilegeEscalation
  - readOnlyRootFilesystem

Security

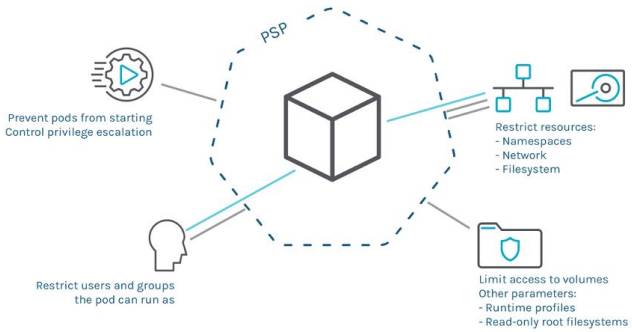
- Example: Pods security context

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: alpineweb
  namespace: namespace-psa
  labels:
    name: alpineweb
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  containers:
    - name: alpine
      image: labdocker/alpineweb:restrict
      resources:
        requests:
          memory: "16Mi"
          cpu: "100m"
        limits:
          memory: "32Mi"
          cpu: "200m"
      ports:
        - containerPort: 22
          protocol: TCP
```

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: alpineweb
  namespace: namespace-psa
  labels:
    name: alpineweb
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  containers:
    - name: alpineweb
      image: labdocker/alpineweb:restrict
      resources:
        requests:
          memory: "16Mi"
          cpu: "100m"
        limits:
          memory: "32Mi"
          cpu: "200m"
      ports:
        - containerPort: 22
          protocol: TCP
      volumeMounts:
      - mountPath: /temp
        name: voldocker
  volumes:
    - name: voldocker
      hostPath:
        path: /var/lib/containerd/
```

Workshop: Security

- Part: Pod Security Association (PSA)



PodSecurity admission (PodSecurityPolicy replacement) #2579

Open | 4 of 8 tasks | tallclair opened this issue on Mar 20 · 14 comments

tallclair commented on Mar 20 • edited by liggitt — Member

Enhancement Description

- One-line enhancement description (can be used as a release note): Introduce a new admission controller for enforcing the Pod Security Standards on pods in a namespace.
- Kubernetes Enhancement Proposal: https://github.com/kubernetes/enhancements/blob/master/keps/sig-auth/2579-psp-replacement/
- Discussion Link: https://docs.google.com/document/d/1woLGRoONE3EBVx-wTb4pvp4CI7tmLZ6lS26VTbosLKM/view#bookmark=id.km06bp3uzuco
- Primary contact (assignee): @tallclair
- Responsible SIGs: sig-auth, sig-security
- Enhancement target (which target equals to which milestone):
  - Alpha release target (x.y): 1.22
  - Beta release target (x.y): 1.23
  - Stable release target (x.y):

☑ Alpha

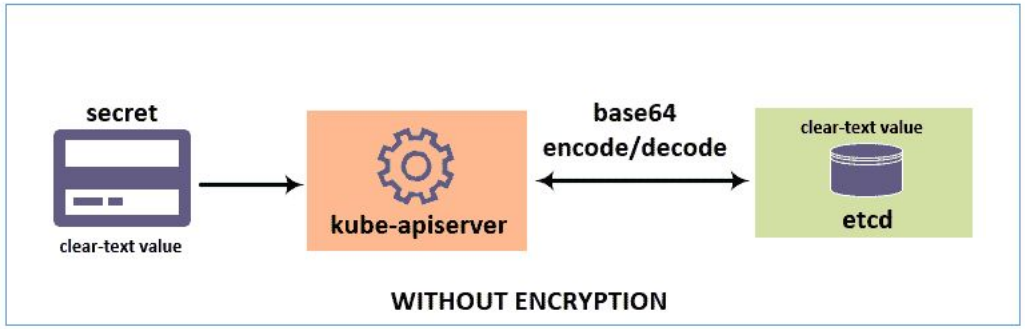
- ☑ KEP (`k/enhancements`) update PR(s): PSP Replacement KEP #2582
- ☑ Code (`k/k`) update PR(s):
  - [PodSecurity] Add ValidatePodSecurityConfiguration kubernetes#103560
  - PodSecurity message/check/fixture cleanups kubernetes#103558
  - PodSecurity: use code/reason/details from admission library kubernetes#103552
  - Implement check drop capabilities.go kubernetes#103543
  - Podsecurity fixture cleanup kubernetes#103517
  - Podsecurity webhook kubernetes#103465
  - Move pod-security-admission to an external Attributes interface kubernetes#103445
  - [PodSecurity] hostProcess baseline check kubernetes#103382
  - [PodSecurity] baseline - apparmor kubernetes#103378
  - PodSecurity: make failure integration tests feature-aware kubernetes#103365
  - [PodSecurity] Add privileged containers baseline check kubernetes#103364

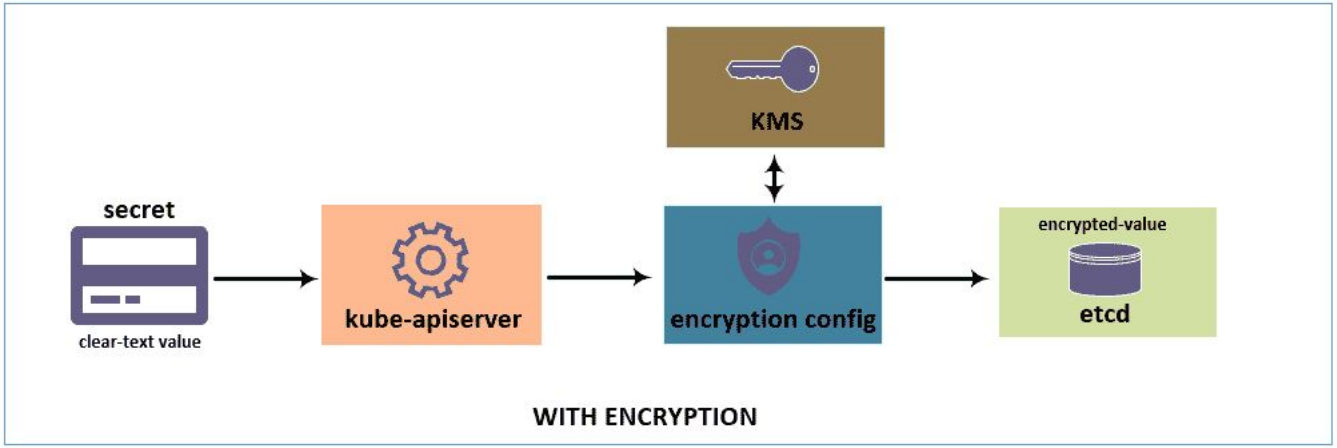
Security

- Encryption Provider: Make secret great again
  - Kubernetes have default secret for encryption sensitive information and hide value from view with describe command etc.
  - Anyway as basic secret is encryption via base64 method. That mean on technical term. It possible to reverse engineering for decode64 from plan yaml (Aka store data as plaintext on etcd datastore).
  - For this approach many user will select alternative option for manage this (Ex: Vault at most etc.)
  - Kubernetes SIG also got this concern and provide solution to "encrypt" data on etcd again for make it more secure.
  - Kubernetes provide a lot of encryption provider for do this job (Ex: AESCBC, AESGCM etc.) and start from version 1.27 (KMS v1) have option to encrypt secret/configmap/other on etcd with KMS (Key Management System) again for increase security
  - On Kubernetes version 1.28 KMS v1 was announced deprecated and on version 1.29 now had been support KMS v2 (stable) that better security

Security

- Key Management System: For better secret key encryption





Ref:https://4sysops.com/archives/encrypt-kubernetes-secrets-at-rest/

Security

KMS v1

```yaml
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
      - secrets
      - configmaps
      - pandas.awesome.bears.example
    providers:
      - kms:
          name: myKmsPluginFoo
          endpoint: unix:///tmp/socketfile.sock
          cachesize: 100
          timeout: 3s
      - kms:
          name: myKmsPluginBar
          endpoint: unix:///tmp/socketfile.sock
          cachesize: 100
          timeout: 3s
```

KMS v2

```yaml
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
      - secrets
      - configmaps
      - pandas.awesome.bears.example
    providers:
      - kms:
          apiVersion: v2
          name: myKmsPluginFoo
          endpoint: unix:///tmp/socketfile.sock
          timeout: 3s
      - kms:
          apiVersion: v2
          name: myKmsPluginBar
          endpoint: unix:///tmp/socketfile.sock
          timeout: 3s
```

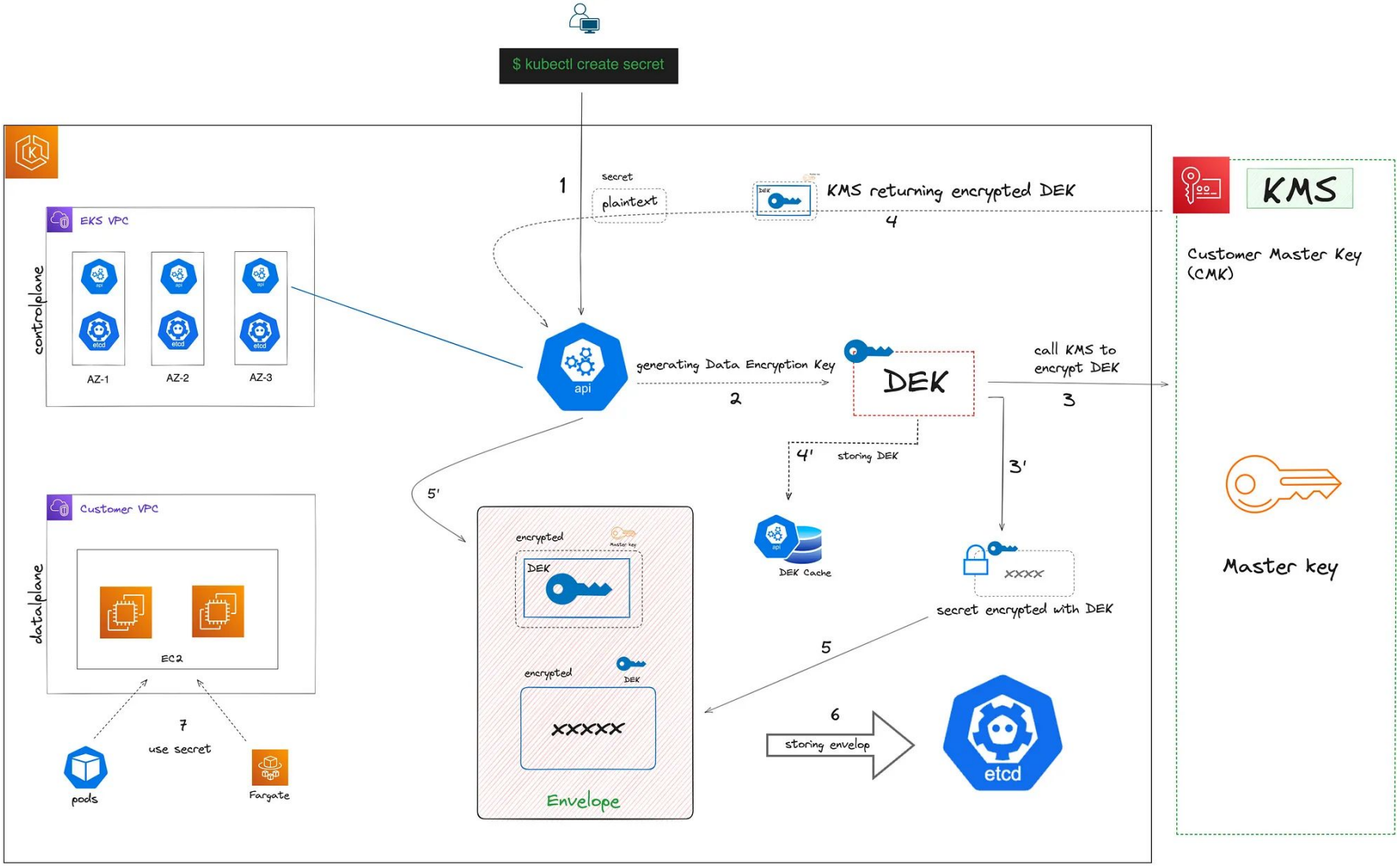
Ref: https://kubernetes.io/docs/tasks/administer-cluster/kms-provider/

Security

Name	Encryption	Strength	Speed	Key length
`identity`	**None**	N/A	N/A	N/A
	Resources written as-is without encryption. When set as the first provider, the resource will be decrypted as new values are written. Existing encrypted resources are **not** automatically overwritten with the plaintext data. The `identity` provider is the default if you do not specify otherwise.			
`aescbc`	AES-CBC with PKCS#7 padding	Weak	Fast	32-byte
	Not recommended due to CBC's vulnerability to padding oracle attacks. Key material accessible from control plane host.			
`aesgcm`	AES-GCM with random nonce	Must be rotated every 200,000 writes	Fastest	16, 24, or 32-byte
	Not recommended for use except when an automated key rotation scheme is implemented. Key material accessible from control plane host.			
`kms` v1 ***(deprecated since Kubernetes v1.28)***	Uses envelope encryption scheme with DEK per resource.	Strongest	Slow (*compared to `kms` version 2*)	32-bytes
	Data is encrypted by data encryption keys (DEKs) using AES-GCM; DEKs are encrypted by key encryption keys (KEKs) according to configuration in Key Management Service (KMS). Simple key rotation, with a new DEK generated for each encryption, and KEK rotation controlled by the user. Read how to configure the KMS V1 provider.			
`kms` v2	Uses envelope encryption scheme with DEK per API server.	Strongest	Fast	32-bytes
	Data is encrypted by data encryption keys (DEKs) using AES-GCM; DEKs are encrypted by key encryption keys (KEKs) according to configuration in Key Management Service (KMS). Kubernetes generates a new DEK per encryption from a secret seed. The seed is rotated whenever the KEK is rotated. A good choice if using a third party tool for key management. Available as stable from Kubernetes v1.29. Read how to configure the KMS V2 provider.			
`secretbox`	XSalsa20 and Poly1305	Strong	Faster	32-byte
	Uses relatively new encryption technologies that may not be considered acceptable in environments that require high levels of review. Key material accessible from control plane host.			

Ref: https://kubernetes.io/docs/tasks/administer-cluster/encrypt-data/

Security



Ref:https://teamoptimizers.medium.com/envelope-encryption-in-eks-4d6006a2ccf

Security

Padding oracle attack

3 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

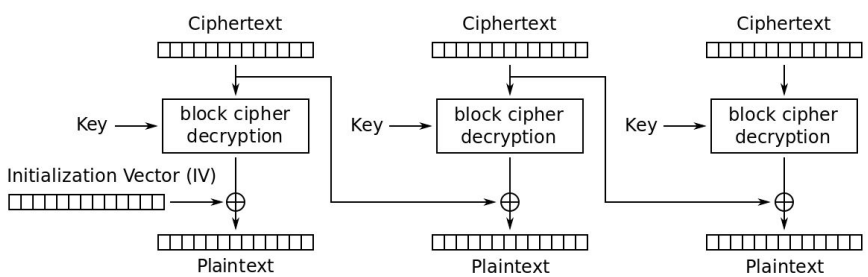
In cryptography, a **padding oracle attack** is an attack which uses the padding validation of a cryptographic message to decrypt the ciphertext. In cryptography, variable-length plaintext messages often have to be padded (expanded) to be compatible with the underlying cryptographic primitive. The attack relies on having a "padding oracle" who freely responds to queries about whether a message is correctly padded or not. Padding oracle attacks are mostly associated with CBC mode decryption used within block ciphers. Padding modes for asymmetric algorithms such as OAEP may also be vulnerable to padding oracle attacks.[1]

Symmetric cryptography [edit]

In symmetric cryptography, the padding oracle attack can be applied to the CBC mode of operation, where the "oracle" (usually a server) leaks data about whether the padding of an encrypted message is correct or not. Such data can allow attackers to decrypt (and sometimes encrypt) messages through the oracle using the oracle's key, without knowing the encryption key.

Padding oracle attack on CBC encryption [edit]

The standard implementation of CBC decryption in block ciphers is to decrypt all ciphertext blocks, validate the padding, remove the PKCS7 padding, and return the message's plaintext. If the server returns an "invalid padding" error instead of a generic "decryption failed" error, the attacker can use the server as a padding oracle to decrypt (and sometimes encrypt) messages.

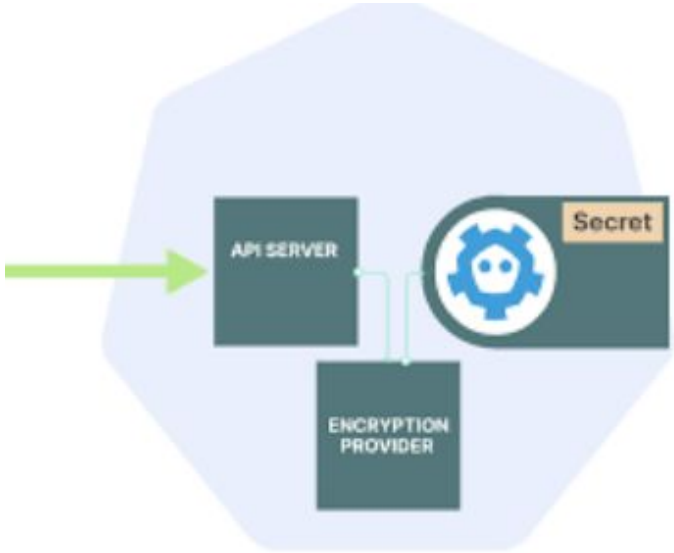


Cipher Block Chaining (CBC) mode decryption

Ref:https://en.wikipedia.org/wiki/Padding_oracle_attack

Workshop: Security

- Part: Encryption Provider




```
surender@kube-srv1:~$ kubectl get secrets --all-namespaces -o json | kubectl replace -f -
secret/app-secret replaced
secret/db-secret replaced
secret/harbor-secret replaced
surender@kube-srv1:~$
surender@kube-srv1:~$ sudo ETCDCTL_API=3 etcdctl \
   --cert=/etc/kubernetes/pki/etcd/server.crt \
   --key=/etc/kubernetes/pki/etcd/server.key \
   --cacert=/etc/kubernetes/pki/etcd/ca.crt \
   get /registry/secrets/default/db-secret | hexdump -C
00000000  2f 72 65 67 69 73 74 72  79 2f 73 65 63 72 65 74  |/registry/secret|
00000010  73 2f 64 65 66 61 75 6c  74 2f 64 62 2d 73 65 63  |s/default/db-sec|
00000020  72 65 74 0a 6b 38 73 3a  65 6e 63 3a 73 65 63 72  |ret.k8s:enc:secr|
00000030  65 74 62 6f 78 3a 76 31  3a 6b 65 79 31 3a 31 94  |etbox:v1:key1:1.|
00000040  75 c3 21 83 a3 e1 e2 ea  e3 18 60 ac 69 22 7a 22  |u.!.......`.i"z"|
00000050  77 41 5f 3f 8c 84 bd 05  28 60 30 a8 af 08 81 5d  |wA_?....(`0....]|
00000060  b2 d7 3b bb ff c4 2b a7  ce 67 7d d6 32 69 2e a4  |..;...+..g}.2i..|
00000070  f9 3f d0 84 b5 32 51 58  3a 90 55 6f 94 90 6f be  |.?...2QX:.Uo..o.|
00000080  c0 0a 52 22 dc c6 01 9a  ea a5 46 a6 75 f0 86 27  |..R".....F.u..'|
00000090  b2 d6 81 9a c5 c8 04 14  6c 77 29 a7 7d 7a c4 a7  |........lw).}z..|
000000a0  e2 36 ab a2 3f 4a 5f f6  1c f7 cf 2a 5f 0d 58 8e  |.6..?J_....*_.X.|
000000b0  f7 ce 9a 34 50 15 ab c6  29 7d 23 82 57 77 69 c0  |...4P...)}#.Wwi.|
000000c0  76 ab 2c 2c c6 c7 87 2a  cf fb d1 24 0b 94 b0 29  |v.,,...*...$...)|
000000d0  58 25 71 01 2b 8d a9 aa  01 f5 b0 aa f3 46 e5 ae  |X%q.+........F..|
000000e0  04 1e c8 10 96 82 95 9f  48 93 31 f2 71 62 ce 28  |........H.1.qb.(|
000000f0  28 be 12 5a ce 9b b2 46  af b9 02 6f 2f 66 ac 90  |(..Z...F...o/f..|
00000100  60 b8 ef 46 8c 9e 31 5c  9d 4e 76 18 a0 89 b2 61  |`..F..1\.Nv....a|
00000110  af 18 49 d1 7b 07 74 a6  87 93 82 2f 9f 3c 82 b2  |..I.{.t..../.<..|
00000120  55 4b d2 5e f9 73 22 92  34 4b 13 1d 45 6f 9c 9d  |UK.^.s".4K..Eo..|
00000130  d6 ef 72 d9 c1 0b 4e 8a  03 58 ef 09 98 b9 f9 c5  |..r...N..X......|
00000140  7b 44 39 c9 b2 a7 d3 02  17 4b 74 da 18 9f 3a 30  |{D9......Kt...:0|
00000150  f1 c9 1f 18 d1 f6 e0 2d  97 4c 5f 80 3a a4 45 94  |.......-.L_.:.E.|
00000160  12 44 1e 90 5e da 36 a9  97 cd 40 e9 c2 6e 81 3c  |.D..^.6...@..n.<|
00000170  69 2d 5a f9 dc 26 12 90  8b 11 c0 82 73 8a 0a     |i-Z..&......s..|
0000017f
surender@kube-srv1:~$
```

← No clear-text value

```yaml
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
      - secrets
    providers:
      - secretbox:
          keys:
            - name: key1
              secret: XXXXXXXXXXXX
      - identity: {} # default encryption provider to allow reading unencrypted secrets
```

Kubernetes in real world

Kubernetes in real world

- Kubernetes have a lot of component need to install / configuration on real world
- Normally kubernetes need some 3rd tool for deployment
  - Ubuntu(MAAS, JUiu)
  - CoreOS
  - Fedora (Ansible)

- provide new solution for create cluster system with kubernetes orchestrator

kubeadm <init>

  - Option:
    - --apiserver-advertise-address= x.x.x.x (Default: First Interface)
    - --apiserver-bind-port = xxx (Default: 6443)
    - --kubernetes-version = xxx (Default: latest)
    - --pod-network-cidr = x.x.x.x (CNI), Need for Flennel
    - --token = xxxx (Default: auto gen)
    - --skip-preflight-checks
    - etc

Kubernetes in real world

- Kubernetes have a lot of component need to install / configuration on real world
- Normally kubernetes need some 3rd tool for deployment
  - Ubuntu(MAAS, JUiu)
  - CoreOS
  - Fedora (Ansible)
- Since kubernetes 1.6 and later. Cluster system can make with command "kubeadm"
- Prerequisite
  - Docker Engine
  - Kubernetes Engine (Kubelet, Kubectl, Kubeadm)
- Role of the cluster system
  - Master:
    - Control all activity on cluster system (Pods, Services, Job, CronJob, RS, RC etc)
  - Node:
    - Worker for running as Master's order

Kubernetes in real world

- Create cluster from bare metal
- Step for setup
  - Phase 1: Install prerequisite component
    - Docker Engine
    - Kubelet Engine
    - Kubeadm Engine
  - Phase 2: Initialize Master node
    - kubeadm init <option>
  - Phase 3: Install Pods Network (3rd party) with CNI support
    - Flannel (Support Cross-Platform)
    - Weave Net (Support Cross-Platform)
    - Calico
    - Canal (Flannel + Calico)
    - Contiv
    - Romana
    - Kube-Router
    - Etc
  - Phase 4: Join node to cluster system
    - kubeadm join <option>

Kubernetes in real world

Kubeadm

Kubeadm is a tool built to provide `kubeadm init` and `kubeadm join` as best-practice "fast paths" for creating Kubernetes clusters.

kubeadm performs the actions necessary to get a minimum viable cluster up and running. By design, it cares only about bootstrapping, not about provisioning machines. Likewise, installing various nice-to-have addons, like the Kubernetes Dashboard, monitoring solutions, and cloud-specific addons, is not in scope.

Instead, we expect higher-level and more tailored tooling to be built on top of kubeadm, and ideally, using kubeadm as the basis of all deployments will make it easier to create conformant clusters.

How to install

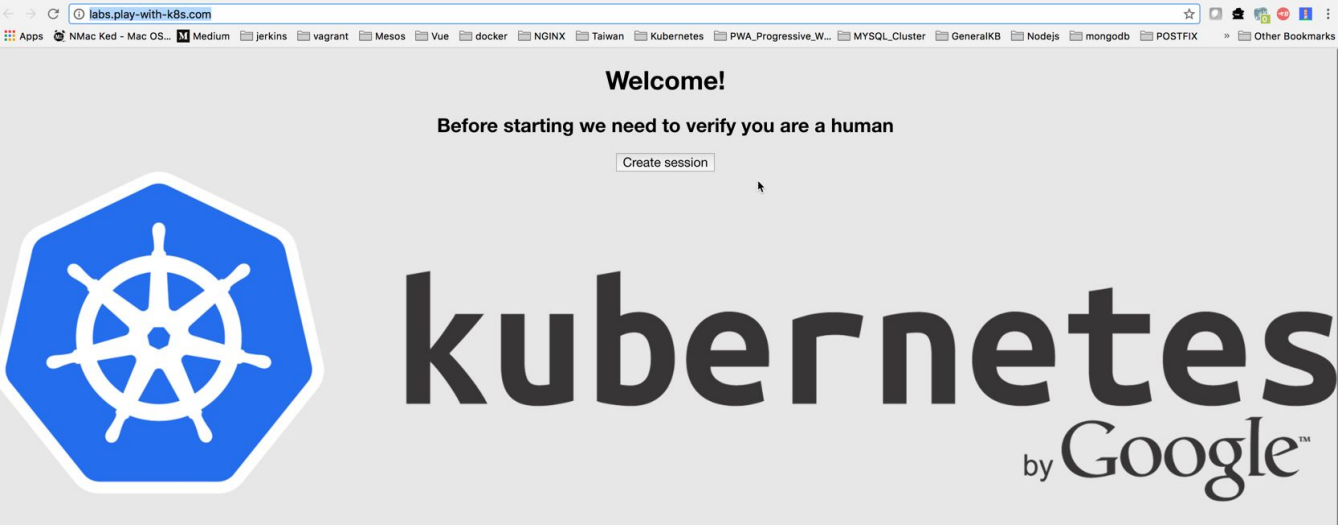
To install kubeadm, see the installation guide.

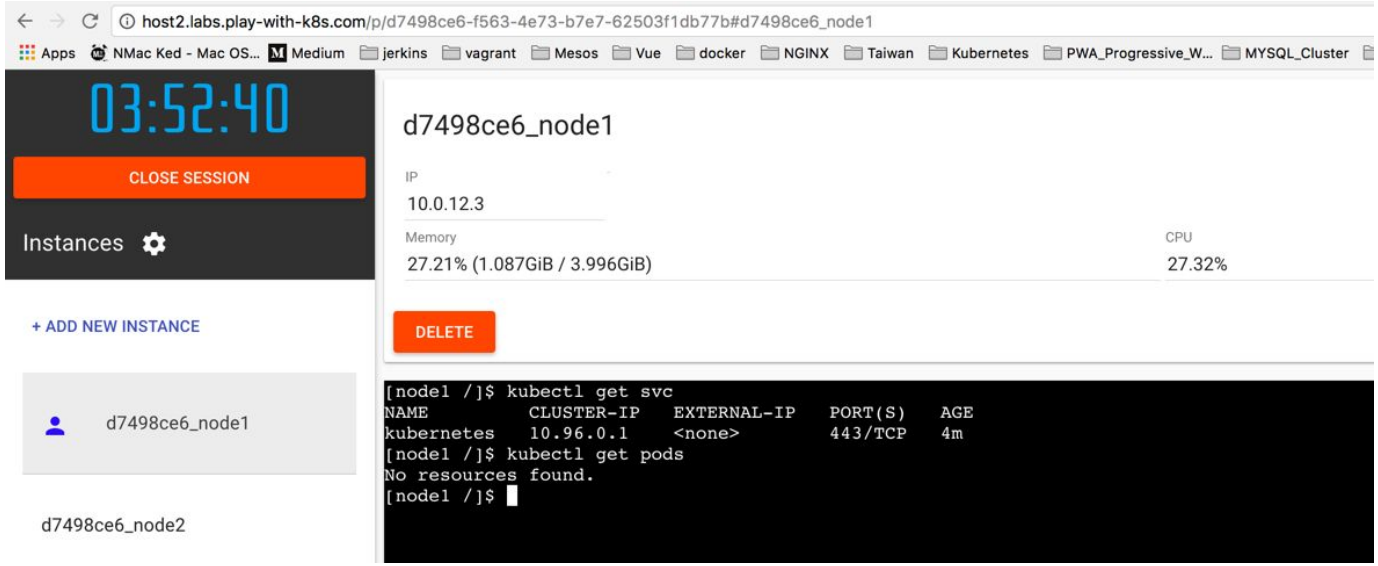
What's next

- kubeadm init to bootstrap a Kubernetes control-plane node
- kubeadm join to bootstrap a Kubernetes worker node and join it to the cluster
- kubeadm upgrade to upgrade a Kubernetes cluster to a newer version
- kubeadm config if you initialized your cluster using kubeadm v1.7.x or lower, to configure your cluster for `kubeadm upgrade`
- kubeadm token to manage tokens for `kubeadm join`
- kubeadm reset to revert any changes made to this host by `kubeadm init` or `kubeadm join`
- kubeadm certs to manage Kubernetes certificates
- kubeadm kubeconfig to manage kubeconfig files
- kubeadm version to print the kubeadm version
- kubeadm alpha to preview a set of features made available for gathering feedback from the community

Ref: https://kubernetes.io/docs/setup/independent/create-cluster-kubeadm/

Kubernetes in real world



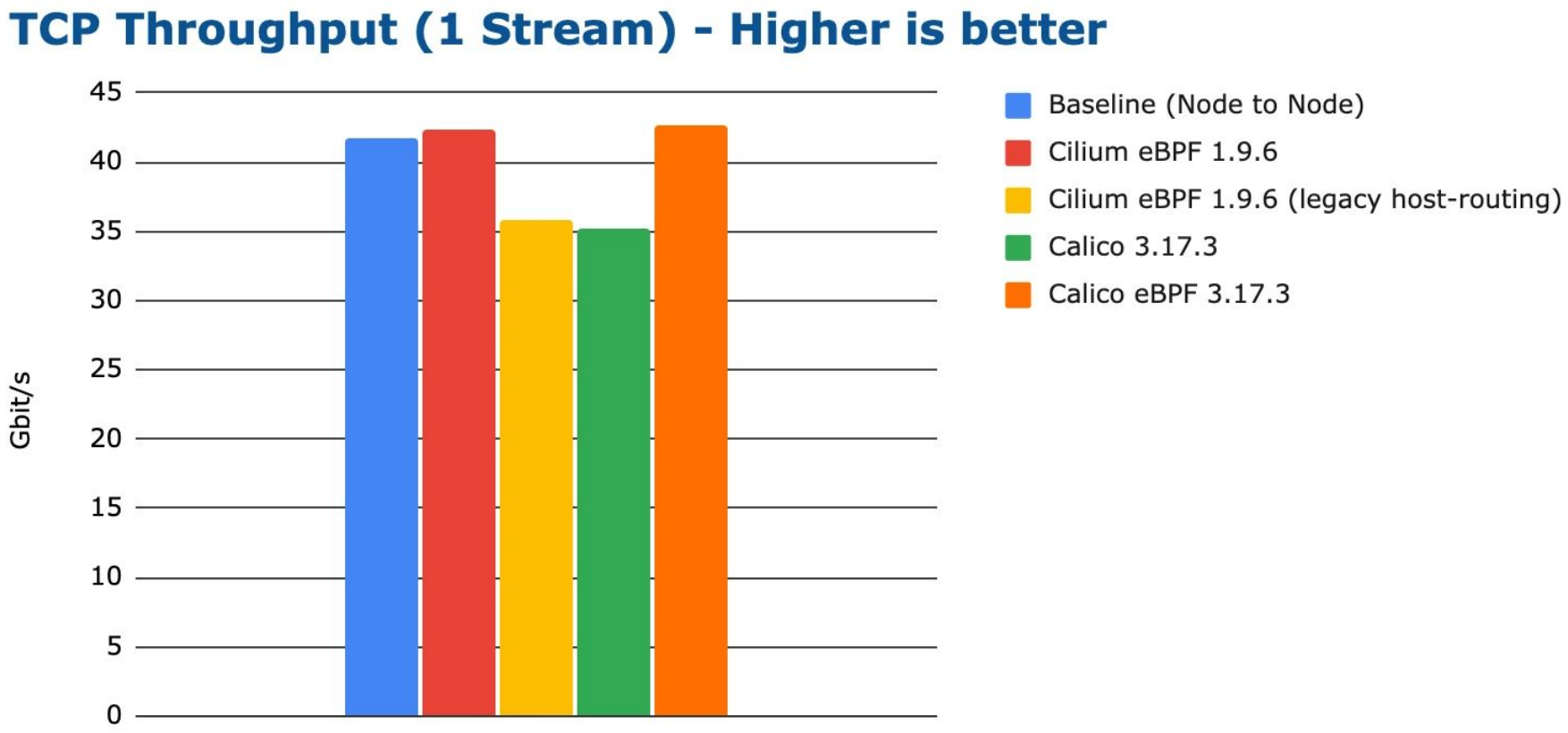


Kubernetes: Production Workload Orchestration

kubernetes by Google

Kubernetes in real world

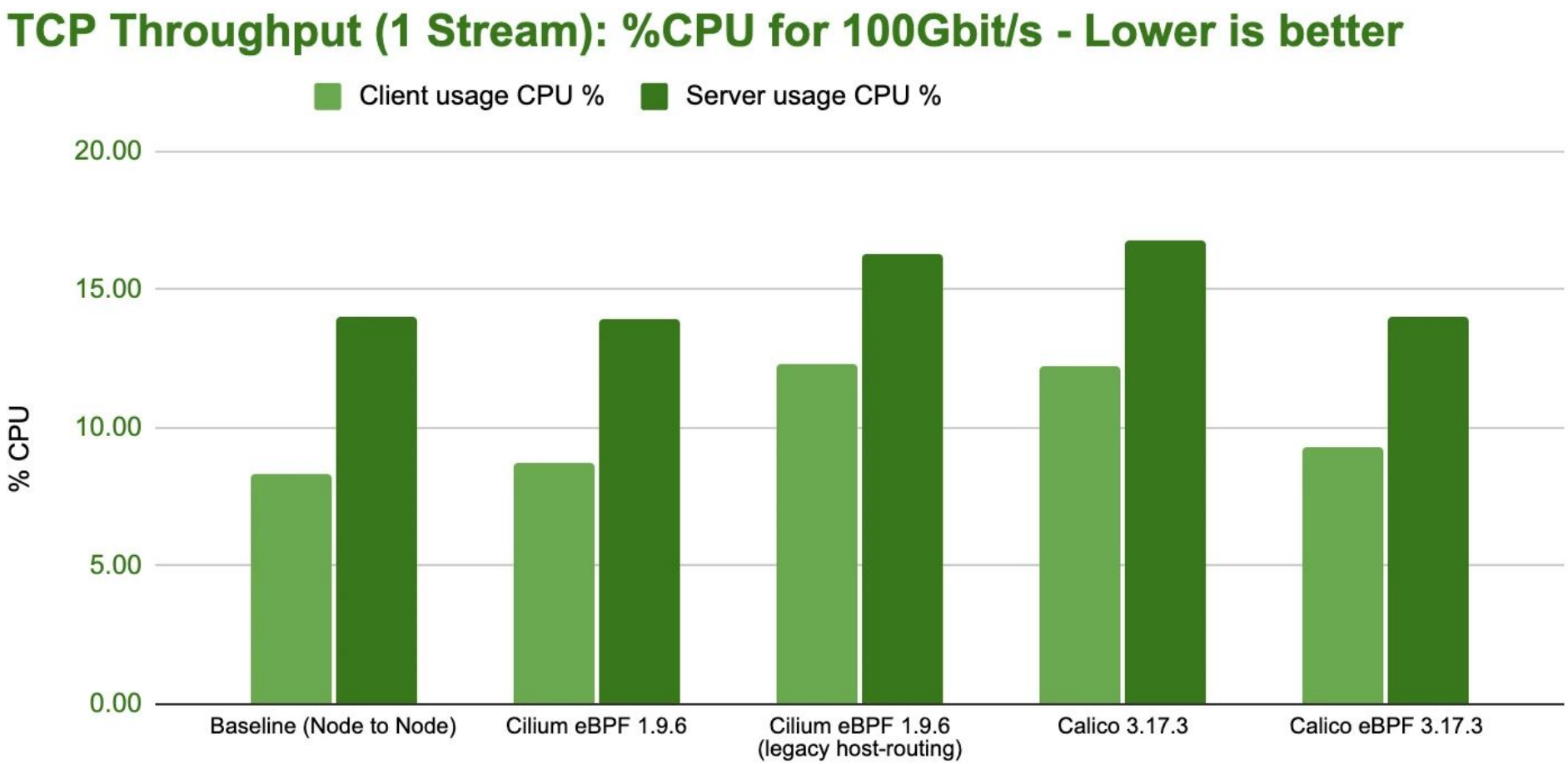
- Network Optional



Ref: https://cilium.io/blog/2021/05/11/cni-benchmark

Kubernetes in real world

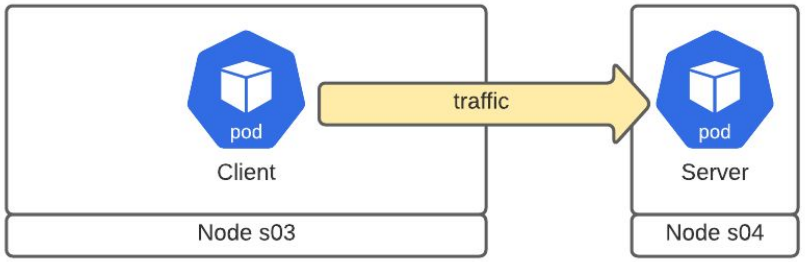
- Network Optional

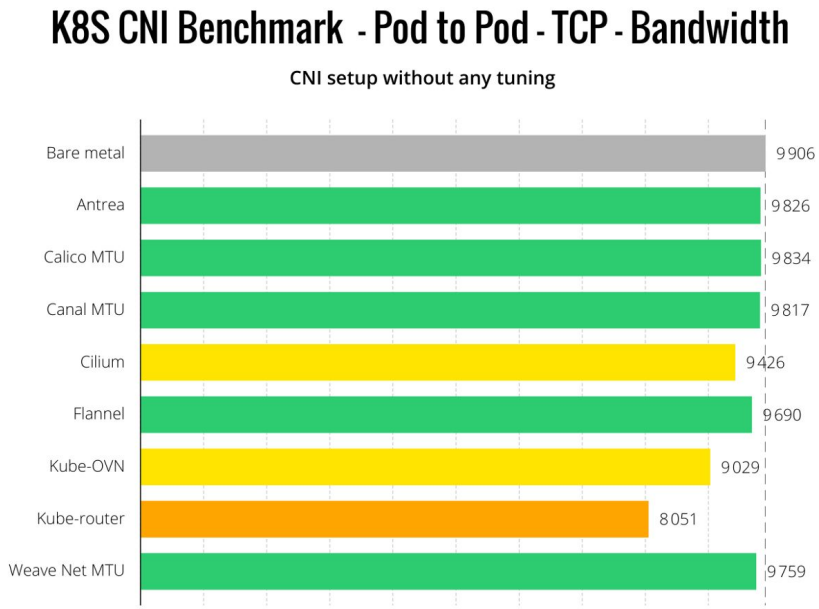


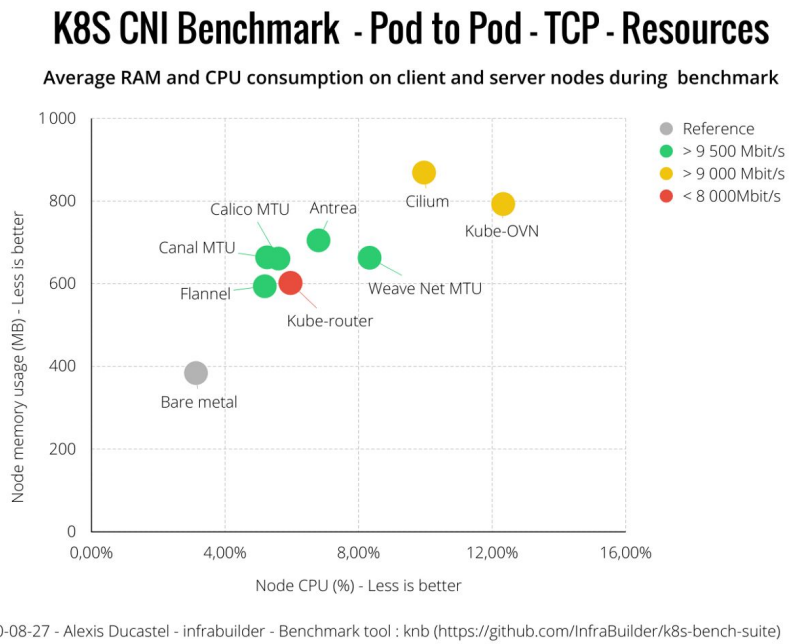
Ref: https://cilium.io/blog/2021/05/11/cni-benchmark

Kubernetes in real world

- Network Optional



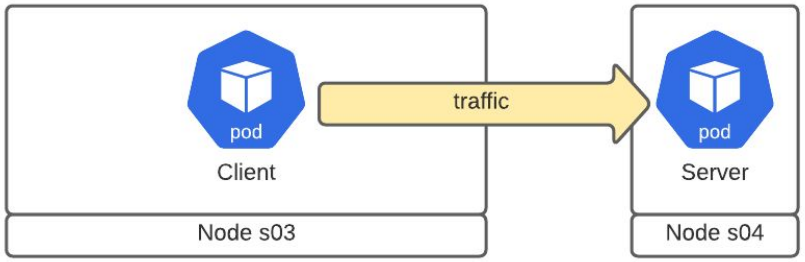


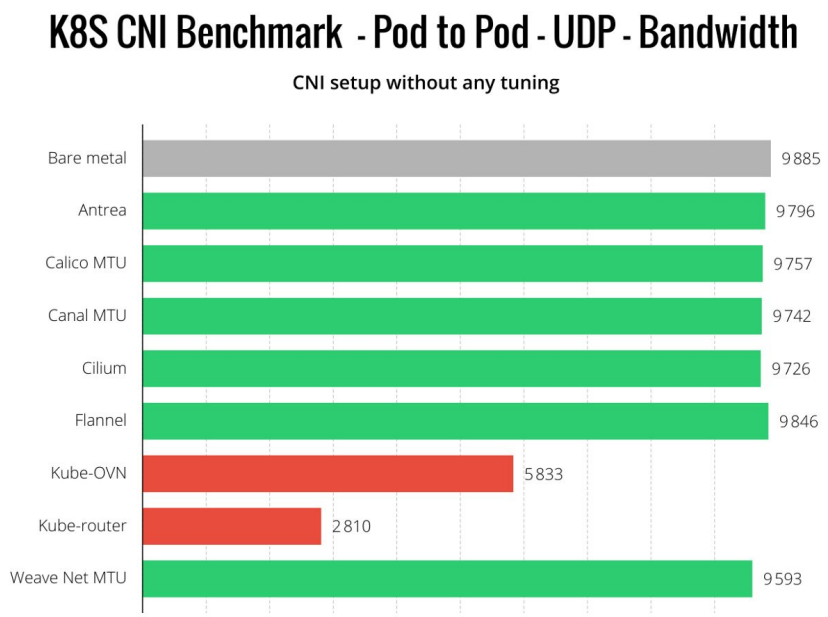


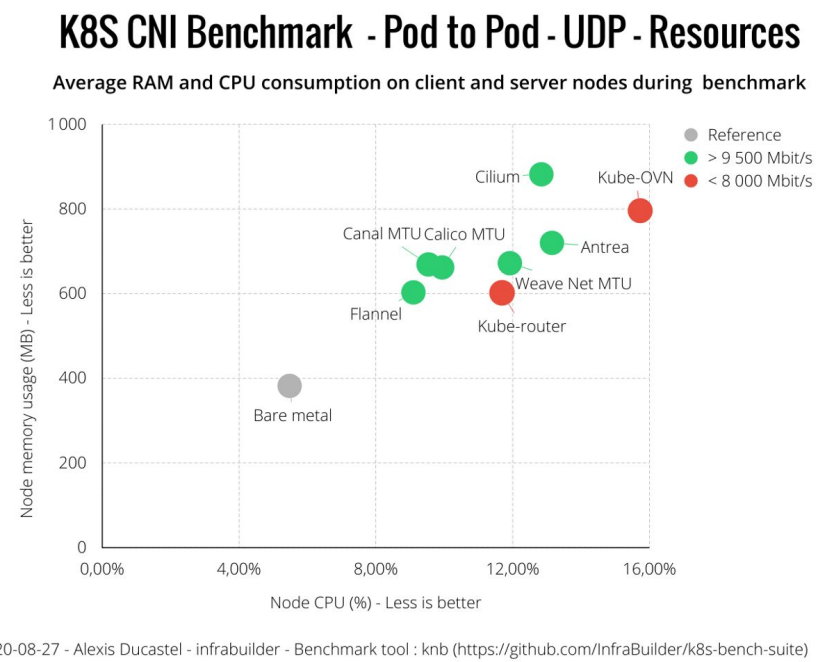
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Network Optional



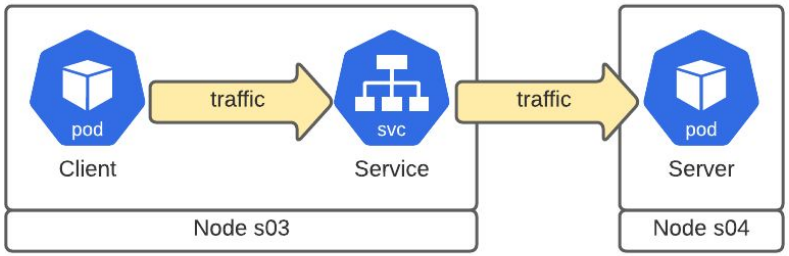


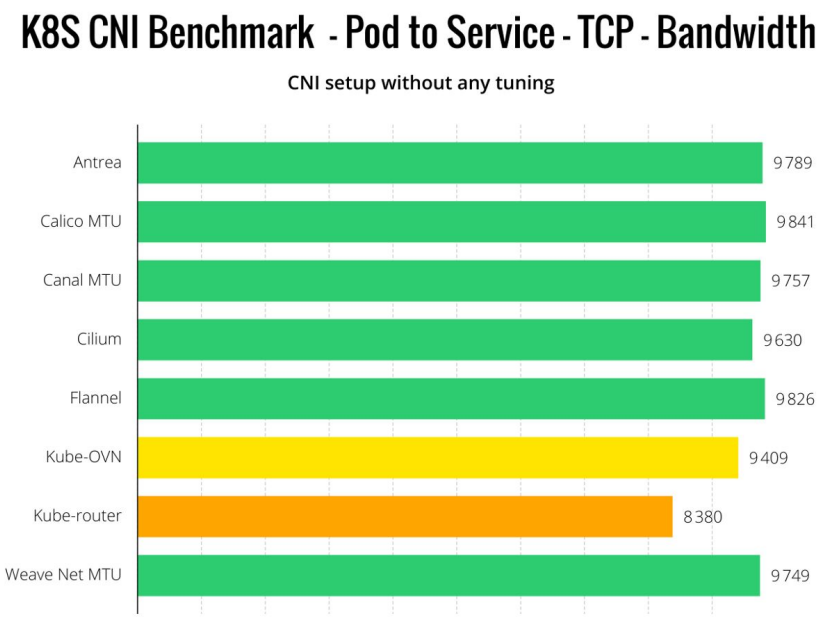


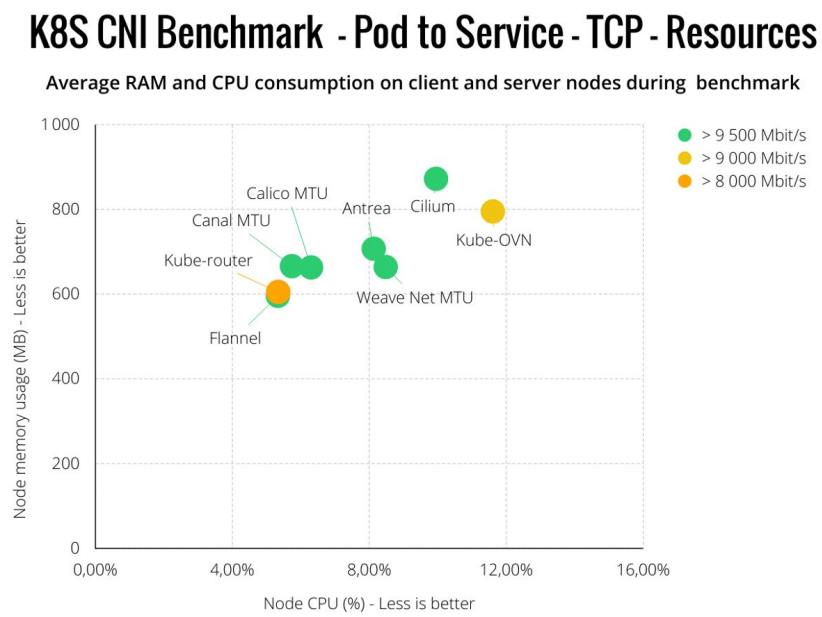
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Network Optional



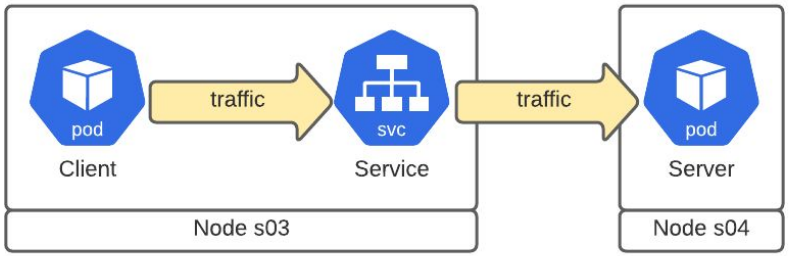


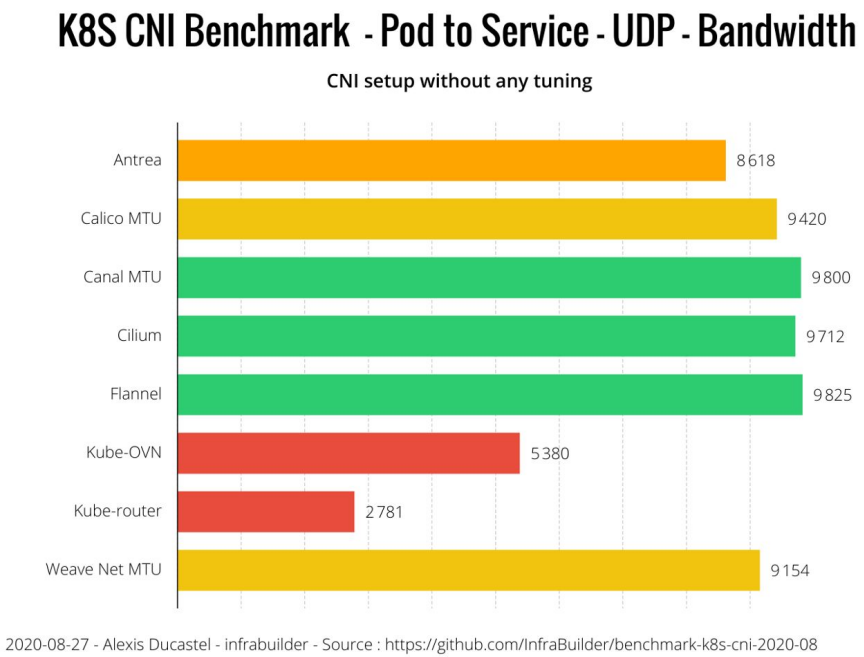


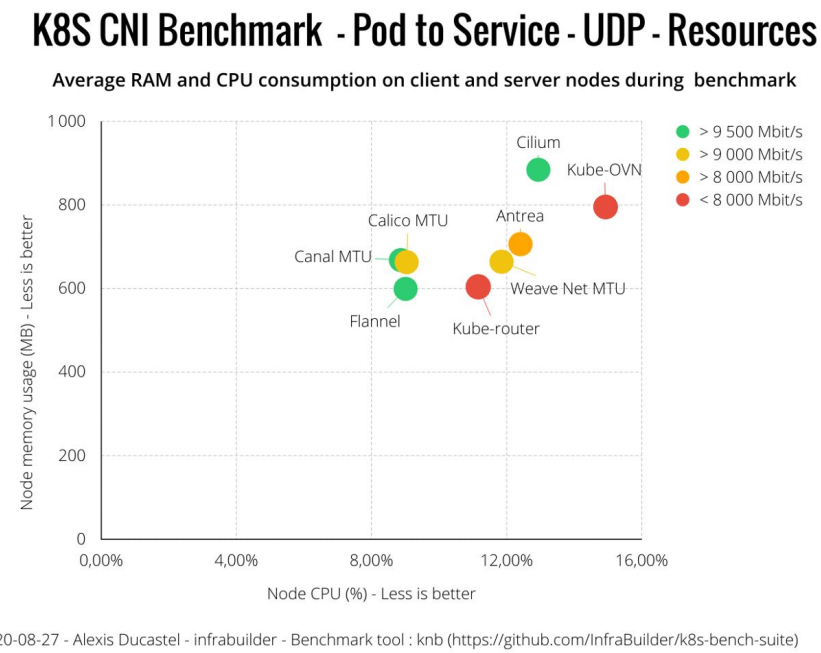
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Network Optional



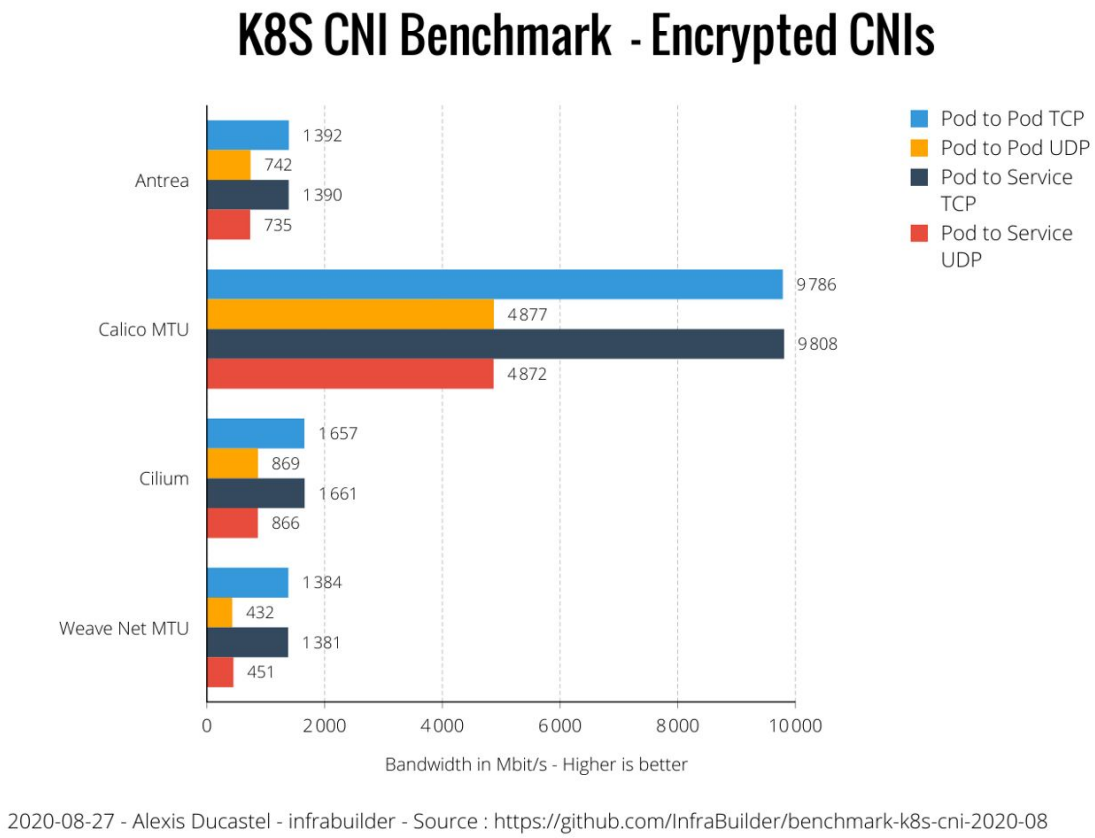




https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

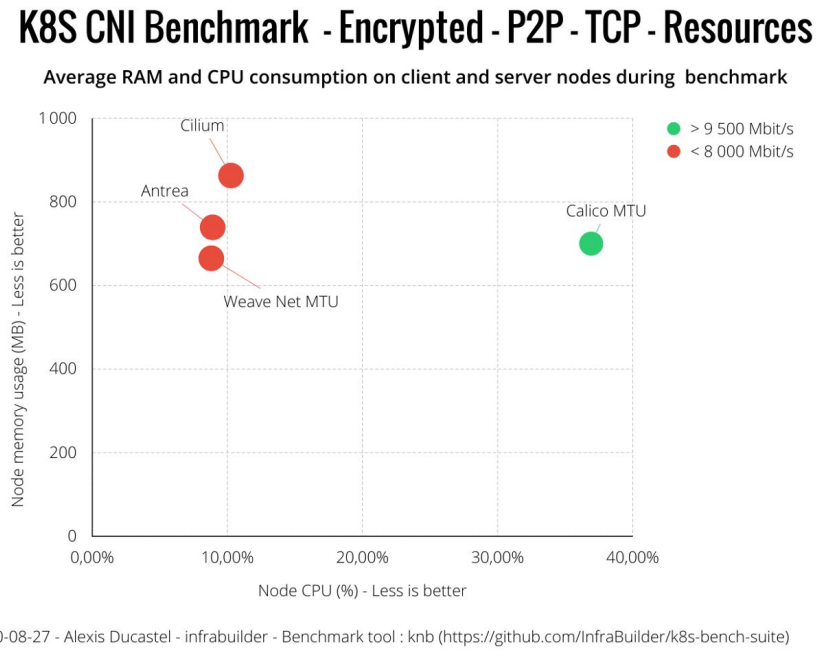
- Network Optional

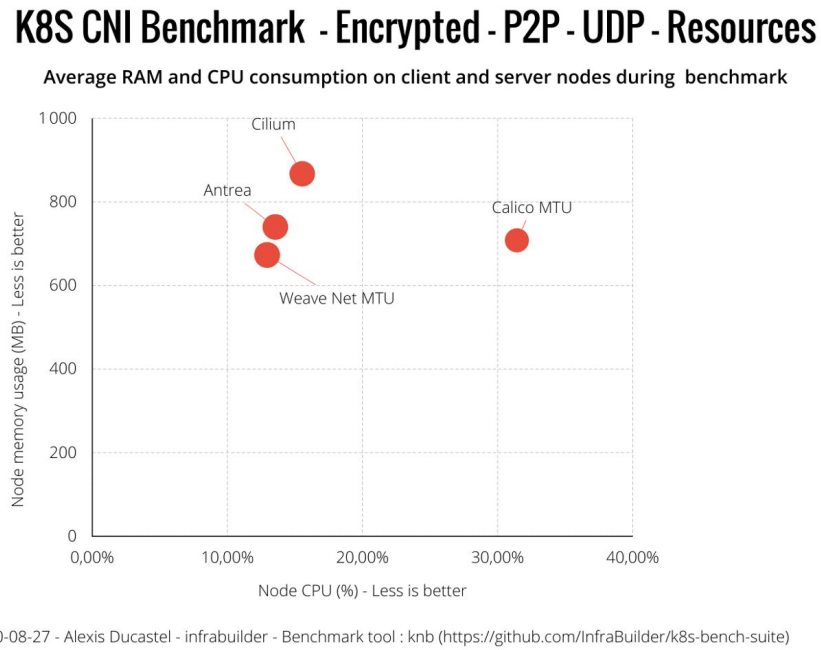


https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

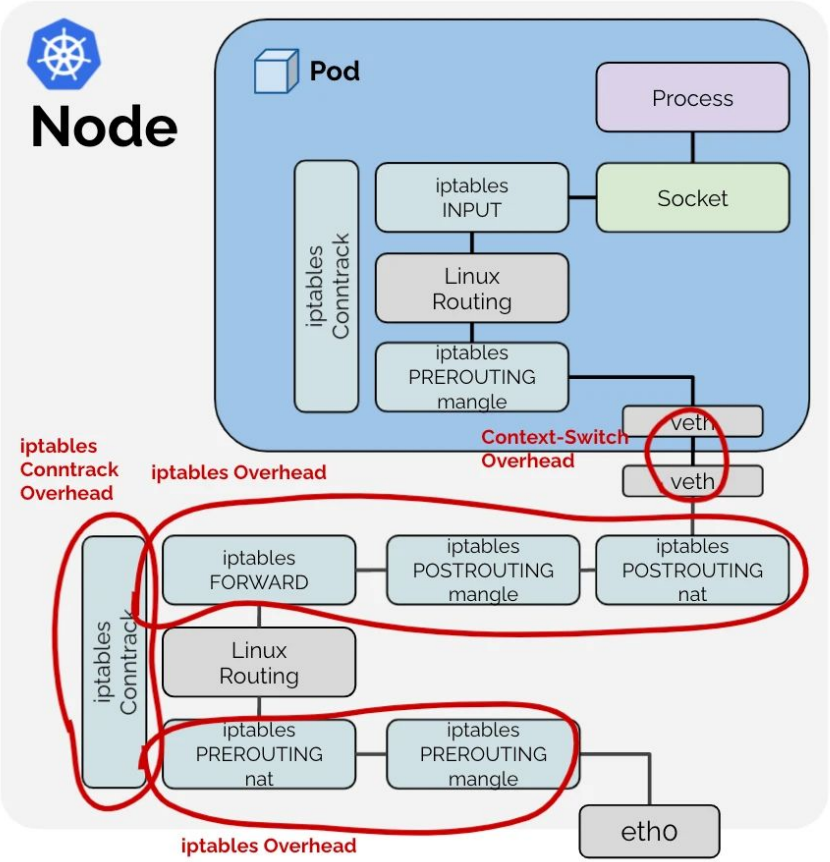
- Network Optional



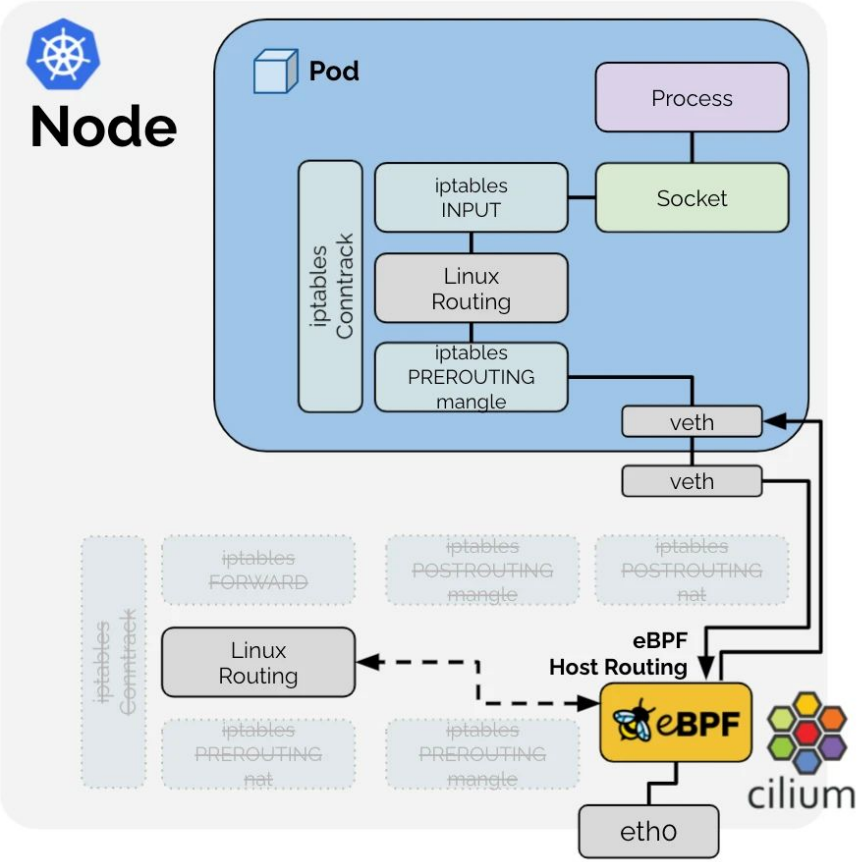


https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world



Standard Container Networking



Cilium eBPF Container Networking

https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Initial Master role

kubeadm <init>

  - Option:
    - --apiserver-advertise-address= x.x.x.x (Default: First Interface)
    - --apiserver-bind-port = xxx (Default: 6443)
    - --kubernetes-version = xxx (Default: latest)
    - --pod-network-cidr = x.x.x.x (CNI), Need for Flennel
    - --token = xxxx (Default: auto gen)
    - --skip-preflight-checks
    - Etc
- Join Node in Cluster

kubeadm <init>

  - Option:
    - --apiserver-advertise-address= x.x.x.x (Default: First Interface)
    - --apiserver-bind-port = xxx (Default: 6443)
    - --kubernetes-version

Kubernetes in real world

- Initial Master role by config file

kubeadm init –config <configuration file>

Untitled-2 ● instruction.txt ! kubeadm-init.yaml ×

```yaml
apiVersion: kubeadm.k8s.io/v1beta1
bootstrapTokens:
- groups:
  - system:bootstrappers:kubeadm:default-node-token
  ttl: 24h0m0s
  usages:
  - signing
  - authentication
kind: InitConfiguration
localAPIEndpoint:
  bindPort: 6443
nodeRegistration:
  criSocket: /var/run/dockershim.sock
  name: hostnamemaster
  kubeletExtraArgs:
    cloud-provider: aws
  taints:
  - effect: NoSchedule
    key: node-role.kubernetes.io/master
---
```

```yaml
---
apiServer:
  certSANs:
  - 1.1.1.1
  extraArgs:
    authorization-mode: Node,RBAC
  timeoutForControlPlane: 4m0s

apiVersion: kubeadm.k8s.io/v1beta1
certificatesDir: /etc/kubernetes/pki
clusterName: Kubernetes
controlPlaneEndpoint: ""
controllerManager:
  extraArgs:
    cloud-provider: aws
    configure-cloud-routes: "false"
    address: 0.0.0.0
dns:
  type: CoreDNS
etcd:
  local:
    dataDir: /var/lib/etcd
imageRepository: k8s.gcr.io
kind: ClusterConfiguration
kubernetesVersion: v1.13.0
networking:
  dnsDomain: cluster.local
  podSubnet: 192.168.0.0/16
  serviceSubnet: 10.96.0.0/12
scheduler:
  extraArgs:
    address: 0.0.0.0
```

Ref: https://godoc.org/k8s.io/kubernetes/cmd/kubeadm/app/apis/kubeadm/v1beta1

Kubernetes in real world

- Join Worker role by config file

kubeadm join –config <configuration file>

Untitled-2 ● instruction.txt ! kubeadm-join.yaml ×

```yaml
apiVersion: kubeadm.k8s.io/v1beta1
caCertPath: /etc/kubernetes/pki/ca.crt
discovery:
  bootstrapToken:
    token: tokenid
    apiServerEndpoint: hostnamemaster:6443
    caCertHashes:
      - "sha256:cahash"
  timeout: 5m0s
kind: JoinConfiguration
nodeRegistration:
  name: hostnameworker
  kubeletExtraArgs:
    cloud-provider: aws
```

Ref: https://godoc.org/k8s.io/kubernetes/cmd/kubeadm/app/apis/kubeadm/v1beta1

Kubernetes in real world

```
  apiVersion: kubeadm.k8s.io/v1beta3
  kind: InitConfiguration
  bootstrapTokens:
  - token: "9a08jv.c0izixklcxtmnze7"
    description: "kubeadm bootstrap token"
    ttl: "24h"
  - token: "783bde.3f89s0fje9f38fhf"
    description: "another bootstrap token"
    usages:
    - authentication
    - signing
    groups:
    - system:bootstrappers:kubeadm:default-node-token
  nodeRegistration:
    name: "ec2-10-100-0-1"
    criSocket: "/var/run/dockershim.sock"
    taints:
    - key: "kubeadmNode"
      value: "master"
      effect: "NoSchedule"
    kubeletExtraArgs:
      v: 4
    ignorePreflightErrors:
    - IsPrivilegedUser
  imagePullPolicy: "IfNotPresent"
  localAPIEndpoint:
    advertiseAddress: "10.100.0.1"
    bindPort: 6443
  certificateKey: "e6a2eb8581237ab72a4f494f30285ec12a9694d750b9785706a83bfcbbbd2204"
skipPhases:
- addon/kube-proxy
  ---
  apiVersion: kubeadm.k8s.io/v1beta3
  kind: ClusterConfiguration
  etcd:
    # one of local or external
    local:
      imageRepository: "k8s.gcr.io"
      imageTag: "3.2.24"
      dataDir: "/var/lib/etcd"
      extraArgs:
        listen-client-urls: "http://10.100.0.1:2379"
      serverCertSANs:
      -  "ec2-10-100-0-1.compute-1.amazonaws.com"
```

```
      peerCertSANs:
      - "10.100.0.1"
    # external:
      # endpoints:
      # - "10.100.0.1:2379"
      # - "10.100.0.2:2379"
      # caFile: "/etcd/kubernetes/pki/etcd/etcd-ca.crt"
      # certFile: "/etcd/kubernetes/pki/etcd/etcd.crt"
      # keyFile: "/etcd/kubernetes/pki/etcd/etcd.key"
  networking:
    serviceSubnet: "10.96.0.0/16"
    podSubnet: "10.244.0.0/24"
    dnsDomain: "cluster.local"
  kubernetesVersion: "v1.21.0"
  controlPlaneEndpoint: "10.100.0.1:6443"
  apiServer:
    extraArgs:
      authorization-mode: "Node,RBAC"
    extraVolumes:
    - name: "some-volume"
      hostPath: "/etc/some-path"
      mountPath: "/etc/some-pod-path"
      readOnly: false
      pathType: File
    certSANs:
    - "10.100.1.1"
    - "ec2-10-100-0-1.compute-1.amazonaws.com"
    timeoutForControlPlane: 4m0s
  controllerManager:
    extraArgs:
      "node-cidr-mask-size": "20"
    extraVolumes:
    - name: "some-volume"
      hostPath: "/etc/some-path"
      mountPath: "/etc/some-pod-path"
      readOnly: false
      pathType: File
  scheduler:
    extraArgs:
      address: "10.100.0.1"
    extraVolumes:
    - name: "some-volume"
      hostPath: "/etc/some-path"
      mountPath: "/etc/some-pod-path"
      readOnly: false
      pathType: File
  certificatesDir: "/etc/kubernetes/pki"
```

```
certificatesDir: "/etc/kubernetes/pki"
imageRepository: "k8s.gcr.io"
clusterName: "example-cluster"
---
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
# kubelet specific options here
---
apiVersion: kubeproxy.config.k8s.io/v1alpha1
kind: KubeProxyConfiguration
# kube-proxy specific options here
```

Ref: https://godoc.org/k8s.io/kubernetes/cmd/kubeadm/app/apis/kubeadm/v1beta3

Kubernetes in real world

- Kubeadm support multiple cloud provider
  - AWS
  - Azure
  - CloudStack
  - GCE
  - OpenStack
  - OVirt
  - Proton
  - Vsphere
  - IBM Cloud Kubernetes Service
  - Baidu Cloud Container Engine
  - etc
- Get fully benefit for utilize capability from cloud provider. Ex: Storage dynamic provision, Network load balancer etc

Kubernetes in real world

- Phase 1: Install prerequisite component

```
#Install Base docker-engine
sudo apt-get remove docker docker-engine
sudo apt-get -y install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo apt-key fingerprint 0EBFCD88
sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"
sudo apt-get update
sudo apt-get -y install docker-ce
sudo groupadd docker
sudo usermod -aG docker $USER
sudo systemctl enable docker

#Install Kubernetes Base
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.7.0/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin/kubectl
sudo apt-get update && apt-get install -y apt-transport-https
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
sudo touch /etc/apt/sources.list.d/kubernetes.list
#sudo bash -c 'echo "deb http://apt.kubernetes.io/ kubernetes-xenial-1.7 main" >
/etc/apt/sources.list.d/kubernetes.list'
sudo bash -c 'echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" >
/etc/apt/sources.list.d/kubernetes.list'
sudo apt-get update
sudo apt-get install -y kubelet=1.7.0-00 kubeadm=1.7.0-00
```

Kubernetes in real world

- Phase 2: Initialize Master node

```
praparn@kubernetes-ms:~$ sudo su -
root@kubernetes-ms:~# kubeadm init --kubernetes-version=v1.7.0 --pod-network-cidr=10.244.0.0/16 --toke
n 8c2350.f55343444a6ffc46
[kubeadm] WARNING: kubeadm is in beta, please do not use it for production clusters.
[init] Using Kubernetes version: v1.7.0
[init] Using Authorization modes: [Node RBAC]
[preflight] Running pre-flight checks
[preflight] WARNING: docker version is greater than the most recently validated version. Docker versio
n: 17.06.0-ce. Max validated version: 1.12
[certificates] Generated CA certificate and key.
[certificates] Generated API server certificate and key.
[certificates] API Server serving cert is signed for DNS names [kubernetes-ms kubernetes kubernetes.de
fault kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 192.168.99.200]
[certificates] Generated API server kubelet client certificate and key.
[certificates] Generated service account token signing key and public key.
[certificates] Generated front-proxy CA certificate and key.
[certificates] Generated front-proxy client certificate and key.
[certificates] Valid certificates and keys now exist in "/etc/kubernetes/pki"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/admin.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/kubelet.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/controller-manager.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/scheduler.conf"
[apiclient] Created API client, waiting for the control plane to become ready
```

```
Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run (as a regular user):

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  http://kubernetes.io/docs/admin/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join --token 8c2350.f55343444a6ffc46 192.168.99.200:6443

root@kubernetes-ms:~#
```

Kubernetes in real world

- Phase 3: Install Pods Network (3rd party) with CNI support

```
root@kubeserve-ms: ~ — zsh   praparn@kubernetes-ms: ~ — s...   praparn@kubernetes-1: ~ — ssh...   praparn@kubernetes-2: ~ — ssh...
praparn@kubernetes-ms:~$ kubectl apply -n kube-system -f "https://cloud.weave.works/k8s/net?k8s-versio
n=$(kubectl version | base64 | tr -d '\n')"
serviceaccount "weave-net" created
clusterrole "weave-net" created
clusterrolebinding "weave-net" created
daemonset "weave-net" created
praparn@kubernetes-ms:~$ kubectl get pods --all-namespaces
NAMESPACE     NAME                                    READY     STATUS              RESTARTS   AGE
kube-system   etcd-kubernetes-ms                      1/1       Running             0          2m
kube-system   kube-apiserver-kubernetes-ms            1/1       Running             0          2m
kube-system   kube-controller-manager-kubernetes-ms   1/1       Running             0          2m
kube-system   kube-dns-2425271678-32cjf               0/3       Pending             0          2m
kube-system   kube-proxy-9jtxj                        1/1       Running             0          2m
kube-system   kube-scheduler-kubernetes-ms            1/1       Running             0          2m
kube-system   weave-net-bg346                         0/2       ContainerCreating   0          15s
praparn@kubernetes-ms:~$
```

```
praparn@kubernetes-ms:~$ kubectl get pods --all-namespaces
NAMESPACE     NAME                                    READY     STATUS    RESTARTS   AGE
kube-system   etcd-kubernetes-ms                      1/1       Running   0          2m
kube-system   kube-apiserver-kubernetes-ms            1/1       Running   0          2m
kube-system   kube-controller-manager-kubernetes-ms   1/1       Running   0          2m
kube-system   kube-dns-2425271678-32cjf               3/3       Running   0          2m
kube-system   kube-proxy-9jtxj                        1/1       Running   0          2m
kube-system   kube-scheduler-kubernetes-ms            1/1       Running   0          2m
kube-system   weave-net-bg346                         2/2       Running   0          51s
praparn@kubernetes-ms:~$
```

Kubernetes in real world

- Phase 3: Install Pods Network (3rd party) with CNI support

```
ubuntu@ip-10-21-1-92:~$ cilium install
ℹ️ using Cilium version "v1.11.2"
🔮 Auto-detected cluster name: kubernetes lab
🔮 Auto-detected IPAM mode: cluster-pool
🔑 Created CA in secret cilium-ca
🔑 Generating certificates for Hubble...
🚀 Creating Service accounts...
🚀 Creating Cluster roles...
🚀 Creating ConfigMap for Cilium version 1.11.2...
🚀 Creating Agent DaemonSet...
🚀 Creating Operator Deployment...
⌛ Waiting for Cilium to be installed and ready...
✅ Cilium was successfully installed! Run 'cilium status' to view installation health
ubuntu@ip-10-21-1-92:~$ cilium hubble enable --ui
🔑 Found CA in secret cilium-ca
✨ Patching ConfigMap cilium-config to enable Hubble...
♻️ Restarted Cilium pods
⌛ Waiting for Cilium to become ready before deploying other Hubble component(s)...
🔑 Generating certificates for Relay...
✨ Deploying Relay from quay.io/cilium/hubble-relay:v1.11.2...
✨ Deploying Hubble UI from quay.io/cilium/hubble-ui:v0.8.5 and Hubble UI Backend from quay.io/cilium/hubble-ui-backend:v0.8.5...
⌛ Waiting for Hubble to be installed...
✅ Hubble was successfully enabled!
ubuntu@ip-10-21-1-92:~$ cilium status
    /¯¯\
 /¯¯\__/¯¯\    Cilium:         OK
 \__/¯¯\__/    Operator:       OK
 /¯¯\__/¯¯\    Hubble:         OK
 \__/¯¯\__/    ClusterMesh:    disabled
    \__/

DaemonSet         cilium             Desired: 1, Ready: 1/1, Available: 1/1
Deployment        cilium-operator    Desired: 1, Ready: 1/1, Available: 1/1
Deployment        hubble-relay       Desired: 1, Ready: 1/1, Available: 1/1
Deployment        hubble-ui          Desired: 1, Ready: 1/1, Available: 1/1
Containers:       cilium             Running: 1
                  cilium-operator    Running: 1
                  hubble-relay       Running: 1
                  hubble-ui          Running: 1
Cluster Pods:     4/4 managed by Cilium
Image versions    cilium             quay.io/cilium/cilium:v1.11.2@sha256:4332428fbb528bda32fffe124454458c9b716c86211266d1a03c4ddf695d7f60: 1
                  cilium-operator    quay.io/cilium/operator-generic:v1.11.2@sha256:4c8bea6818ee3e4932f99e9c1d7efa88b8c0f3cd516160caec878406531e45e7: 1
                  hubble-relay       quay.io/cilium/hubble-relay:v1.11.2@sha256:f031f95f3c9ba8962094649c0cc913f90723d553203444c8fb9a591e38873c9d: 1
                  hubble-ui          quay.io/cilium/hubble-ui-backend:v0.8.5@sha256:2bce50cf6c32719d072706f7ceccad654bfa907b2745a496da99610776fe31ed: 1
                  hubble-ui          docker.io/envoyproxy/envoy:v1.18.2@sha256:e8b37c1d75787dd1e712ff389b0d37337dc8a174a63bed9c34ba73359dc67da7: 1
                  hubble-ui          quay.io/cilium/hubble-ui:v0.8.5@sha256:4eaca1ec1741043cfba6066a165b3bf251590cf4ac66371c4f63fbed2224ebb4: 1
ubuntu@ip-10-21-1-92:~$
```

Kubernetes in real world

- Phase 4: Join node to cluster system

```
praparn@kubernetes-1:~$ sudo su -
root@kubernetes-1:~# kubeadm --token 8c2350.f55343444a6ffc46 join 192.168.99.200:6443
[kubeadm] WARNING: kubeadm is in beta, please do not use it for production clusters.
[preflight] Running pre-flight checks
[preflight] WARNING: docker version is greater than the most recently validated version. Docker versio
n: 17.06.0-ce. Max validated version: 1.12
[discovery] Trying to connect to API Server "192.168.99.200:6443"
[discovery] Created cluster-info discovery client, requesting info from "https://192.168.99.200:6443"
[discovery] Cluster info signature and contents are valid, will use API Server "https://192.168.99.200
:6443"
[discovery] Successfully established connection with API Server "192.168.99.200:6443"
[bootstrap] Detected server version: v1.7.0
[bootstrap] The server supports the Certificates API (certificates.k8s.io/v1beta1)
[csr] Created API client to obtain unique certificate for this node, generating keys and certificate s
igning request
[csr] Received signed certificate from the API server, generating KubeConfig...
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/kubelet.conf"

Node join complete:
* Certificate signing request sent to master and response
  received.
* Kubelet informed of new secure connection details.

Run 'kubectl get nodes' on the master to see this machine join.
root@kubernetes-1:~#
```

```
praparn@kubernetes-ms:~$ kubectl get node
NAME            STATUS    AGE       VERSION
kubernetes-1    Ready     5m        v1.7.0
kubernetes-2    Ready     4m        v1.7.0
kubernetes-ms   Ready     11m       v1.7.0
praparn@kubernetes-ms:~$
```

Workshop: Kubernetes Real World

Machine: Kubernetes_MS

kubernetes

Kubectl

Role: Node
IP Address: 10.0.x.x/16
vCPU: 4
Memory: 2 GB

Machine: Kubernetes_1

kubernetes

Role: Node
IP Address: 10.0.x.x/16
vCPU: 4
Memory: 2 GB

Machine: Kubernetes_2

kubernetes

Role: Node
IP Address: 10.0.x.x/16
vCPU: 4
Memory: 2 GB

AWS

Command/Monitor

AWS CLI

Role: Client

Kubernetes: Production Workload Orchestration

kubernetes by Google

Kubernetes in real world

- Multiple Master

kubernetes

Documentation Blog Partners Community Case Studies English v1.16

Getting started

HOME GETTING STARTED CONCEPTS TASKS TUTORIALS REFERENCE CONTRIBUTE

Search

Getting started

- Release notes and version skew
- Learning environment
- Production environment
  - Container runtimes
  - Installing Kubernetes with deployment tools
    - Bootstrapping clusters with kubeadm
      - Installing kubeadm
      - Troubleshooting kubeadm
      - Creating a single control-plane cluster with kubeadm
      - Customizing control plane configuration with kubeadm
      - Options for Highly Available topology
      - Creating Highly Available clusters with kubeadm
      - Set up a High Availability etcd cluster with kubeadm
      - Configuring each kubelet in your

Creating Highly Available clusters with kubeadm

This page explains two different approaches to setting up a highly available Kubernetes cluster using kubeadm:

- With stacked control plane nodes. This approach requires less infrastructure. The etcd members and control plane nodes are co-located.
- With an external etcd cluster. This approach requires more infrastructure. The control plane nodes and etcd members are separated.

Before proceeding, you should carefully consider which approach best meets the needs of your applications and environment. This comparison topic outlines the advantages and disadvantages of each.

If you encounter issues with setting up the HA cluster, please provide us with feedback in the kubeadm issue tracker.

See also The upgrade documentation.

Caution: This page does not address running your cluster on a cloud provider. In a cloud environment, neither approach documented here works with Service objects of type LoadBalancer, or with dynamic PersistentVolumes.

- **Before you begin**
- **First steps for both methods**
- **Stacked control plane and etcd nodes**
- **External etcd nodes**
- **Common tasks after bootstrapping control plane**
- **Manual certificate distribution**

Ref: https://kubernetes.io/docs/setup/independent/high-availability/

kubernetes by Google

Orchestrator Assignment

Kubernetes: Production Workload Orchestration

kubernetes by Google

Orchestrator Assignment

- Kubernetes have several way for control/restrict pod to run on nodes
- nodeSelector
  - Give some customize label for classify host and define some selector criteria for specific node run Pods
- Interlude
  - Build-in labels with nodes
    - kubernetes.io/hostname
    - failure-domain.beta.kubernetes.io/zone
    - failure-domain.beta.kubernetes.io/region
    - beta.kubernetes.io/instance-type
    - beta.kubernetes.io/arch
- Affinity
  - Node Affinity
  - Inter-Pods Affinity and Anti-Affinity
- Taints and tolerations

nodeSelector

Nodes: kubernetes-ms, kubernetes-1, kubernetes-2

```
praparn@kubernetes-ms:~$ kubectl label nodes kubernetes-ms storage=M2
node "kubernetes-ms" labeled
praparn@kubernetes-ms:~$ kubectl label nodes kubernetes-1 storage=SSD
node "kubernetes-1" labeled
praparn@kubernetes-ms:~$ kubectl label nodes kubernetes-2 storage=SAS
node "kubernetes-2" labeled
praparn@kubernetes-ms:~$ kubectl describe nodes
Name:                   kubernetes-1
Role:
Labels:                 beta.kubernetes.io/arch=amd64
                        beta.kubernetes.io/os=linux
                        kubernetes.io/hostname=kubernetes-1
                        storage=SSD
```

```
Name:                   kubernetes-2
Role:
Labels:                 beta.kubernetes.io/arch=amd64
                        beta.kubernetes.io/os=linux
                        kubernetes.io/hostname=kubernetes-2
                        storage=SAS
```

```
Name:                   kubernetes-ms
Role:
Labels:                 beta.kubernetes.io/arch=amd64
                        beta.kubernetes.io/os=linux
                        kubernetes.io/hostname=kubernetes-ms
                        node-role.kubernetes.io/master=
                        storage=M2
```

Pods YAML File:

```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
  nodeSelector:
    storage: M2
```

nodeSelector

```
praparn@kubernetes-ms:~$ kubectl create -f webtest_pod_nodeselector.yml
pod "webtest" created
praparn@kubernetes-ms:~$ kubectl describe pods/webtest
Name:           webtest
Namespace:      default
Node:           kubernetes-ms/192.168.99.200
Start Time:     Sun, 23 Jul 2017 13:20:34 +0000
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
Annotations:    <none>
Status:         Pending
```

```
Node-Selectors: storage=M2
Tolerations:    node.alpha.kubernetes.io/notReady:NoExecute for 300s
                node.alpha.kubernetes.io/unreachable:NoExecute for 300s
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type R
eason                   Message
  ---------     --------        -----   ----                    -------------                   ------
--      ------                  -------
  12m           12m             1       default-scheduler                                       NormaS
cheduled                Successfully assigned webtest to kubernetes-ms
  12m           12m             1       kubelet, kubernetes-ms                                  NormaS
uccessfulMountVolume    MountVolume.SetUp succeeded for volume "default-token-1ltqv"
  12m           12m             1       kubelet, kubernetes-ms  spec.containers{webtest}        NormaP
ulling                  pulling image "labdocker/cluster:webservicelite"
  11m           11m             1       kubelet, kubernetes-ms  spec.containers{webtest}        NormaP
ulled                   Successfully pulled image "labdocker/cluster:webservicelite"
  11m           11m             1       kubelet, kubernetes-ms  spec.containers{webtest}        NormaC
reated                  Created container
  11m           11m             1       kubelet, kubernetes-ms  spec.containers{webtest}        NormaS
tarted                  Started container
praparn@kubernetes-ms:~$ kubectl get pods -o wide
NAME      READY     STATUS    RESTARTS   AGE       IP          NODE
webtest   1/1       Running   0          12m       10.32.0.3   kubernetes-ms
praparn@kubernetes-ms:~$
```

Interlude

Nodes: kubernetes-ms, kubernetes-1, kubernetes-2

```
praparn@kubernetes-ms:~$ kubectl label nodes kubernetes-ms storage=M2
node "kubernetes-ms" labeled
praparn@kubernetes-ms:~$ kubectl label nodes kubernetes-1 storage=SSD
node "kubernetes-1" labeled
praparn@kubernetes-ms:~$ kubectl label nodes kubernetes-2 storage=SAS
node "kubernetes-2" labeled
praparn@kubernetes-ms:~$ kubectl describe nodes
Name:                   kubernetes-1
Role:
Labels:                 beta.kubernetes.io/arch=amd64
                        beta.kubernetes.io/os=linux
                        kubernetes.io/hostname=kubernetes-1
                        storage=SSD
```

```
Name:                   kubernetes-2
Role:
Labels:                 beta.kubernetes.io/arch=amd64
                        beta.kubernetes.io/os=linux
                        kubernetes.io/hostname=kubernetes-2
                        storage=SAS
```

```
Name:                   kubernetes-ms
Role:
Labels:                 beta.kubernetes.io/arch=amd64
                        beta.kubernetes.io/os=linux
                        kubernetes.io/hostname=kubernetes-ms
                        node-role.kubernetes.io/master=
                        storage=M2
```

Pods YAML File:

```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
  nodeSelector:
    kubernetes.io/hostname: kubernetes-1
```

Interlude

```
praparn@kubernetes-ms:~$ kubectl create -f webtest_pod_interlude.yml
pod "webtest" created
praparn@kubernetes-ms:~$ kubectl get pods -o wide
NAME      READY     STATUS              RESTARTS   AGE       IP        NODE
webtest   0/1       ContainerCreating   0          6s        <none>    kubernetes-1
praparn@kubernetes-ms:~$ kubectl describe pods/webtest
Name:           webtest
Namespace:      default
Node:           kubernetes-1/192.168.99.201
Start Time:     Sun, 23 Jul 2017 14:11:36 +0000
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
Annotations:    <none>
Status:         Pending
IP:
Containers:
  webtest:
    Container ID:
    Image:              labdocker/cluster:webservicelite
    Image ID:
    Port:               5000/TCP
    State:              Waiting
      Reason:           ContainerCreating
    Ready:              False
    Restart Count:      0
    Environment:        <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-1ltqv (ro)
Conditions:
  Type          Status
  Initialized   True
  Ready         False
  PodScheduled  True
Volumes:
  default-token-1ltqv:
    Type:       Secret (a volume populated by a Secret)
    SecretName: default-token-1ltqv
    Optional:   false
QoS Class:      BestEffort
Node-Selectors: kubernetes.io/hostname=kubernetes-1
```

Affinity

- Node Affinity
  - Similar to “nodeSelector” but more flexible for selection
    - requiredDuringSchedulingIgnoredDuringExecution (Hard)
    - preferredDuringSchedulingIgnoredDuringExecution (Soft)
  - If Pods was define both “nodeSelector” and “Affinity”. It need to satisfy both for operate
- Inter-pod Affinity and Anti-affinity
  - “based on labels on pods (X) that are already running on the node with criteria (Y)"
  - Add constrain/dependence from Pods
    - podAffinity (Run with existing pods)
    - podAntiAffinity (Not run with existing pods)
  - Criteria(Y) will consider as “topologyKey”
  - Ex:
    - Run pods (podAffinity) on any node with same hostname (topologyKey (Y)=hostname) with existing pods that label “environment=development” (X)

Node Affinity

Pods YAML File:

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: beta.kubernetes.io/os
            operator: In #In, NotIn, Exists, DoesNotExist, Gt, Lt
            values:
            - linux
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: storage
            operator: In
            values:
            - SSD
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite
      ports:
        - containerPort: 5000
          protocol: TCP
```

Inter-pod Affinity and Anti-affinity

Pods YAML File:

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest2
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  affinity:
    podAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: environment
            operator: In
            values:
            - development
        topologyKey: kubernetes.io/hostname
    podAntiAffinity:
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        podAffinityTerm:
          labelSelector:
            matchExpressions:
            - key: module
              operator: In
              values:
              - DBServer
          topologyKey: storage
  containers:
    - name: webtest2
      image: labdocker/cluster:webservicelite
      ports:
        - containerPort: 5000
```

"Run Pods on any node that have existing pods with key (environment=development) on node same hostname"

"Don't run Pods on any node that have existing pods with key (module=DBServer) on node same storage type"

Taint and Tolerations

- Node side consideration
- Force/Repel Pods from Node
- Taint will apply to Node for protect “node” to run any pods without “tolerate” match taint
- Tolerations apply to Pods for suggest (Not required) Pods to schedule
- Use Case:
  - Dedicated Node / Maintenance Node
  - Special Hardware Node
- Value for Taint
  - PreferNoSchedule
  - NoSchedule
  - NoExecute

Taint and Tolerations

Taint on Node:

```
praparn@kubernetes-ms:~$ kubectl taint nodes kubernetes-1 dedicated=admin:NoSchedule
node "kubernetes-1" tainted
```

Pods YAML File:

```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
  tolerations:
  - key: "dedicated"
    operator: "Equal"
    value: "admin"
    effect: "NoSchedule"
  nodeSelector:
     kubernetes.io/hostname: kubernetes-1
```

Tolerations:
Key="dedicated"
Operator="Equal"
Value="Admin"
Effect="NoSchedule"

Workshop: Orchestrator Assignment

Machine: Kubernetes_MS

kubernetes

Kubectl

Role: Node
IP Address: 10.0.x.x/16
vCPU: 4
Memory: 2 GB

Label: storage = M2

Machine: Kubernetes_1

kubernetes

Role: Node
IP Address: 10.0.x.x/16
vCPU: 4
Memory: 2 GB

Label: storage = SSD

Machine: Kubernetes_2

kubernetes

Role: Node
IP Address: 10.0.x.x/16
vCPU: 4
Memory: 2 GB

Label: storage = SAS

Command/Monitor

AWS CLI

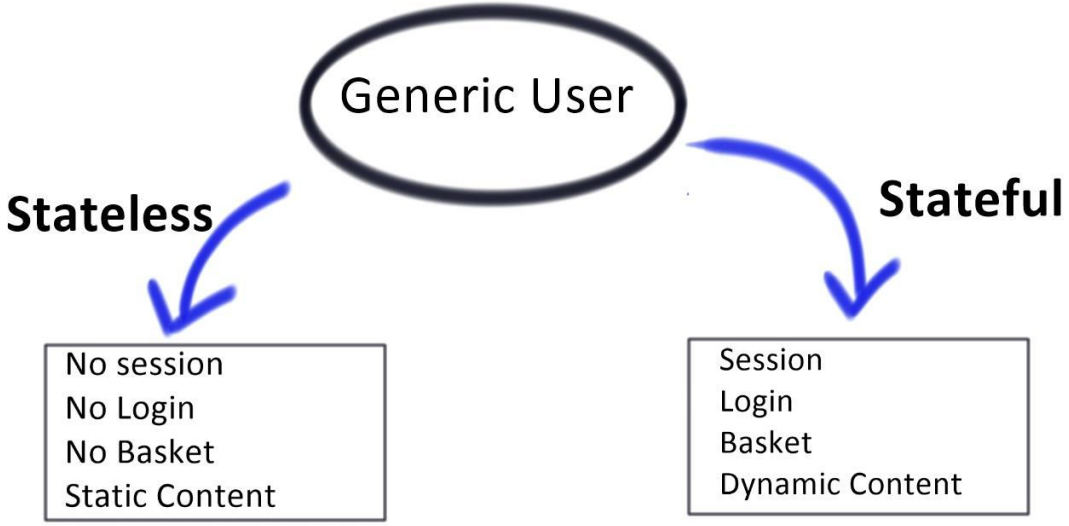
Role: Client

Lab Section:
Part 1: nodeSelector
Part 2: Interlude
Part 3: Affinity (Node)
Part 4: Inter-Pod Affinity and Anti-affinity
Part 5: Taint and Tolerations

Stateful Application Deployment

Stateful Application Deployment

- Some application need "state" for keep application flow and store some information on "session" (Such as login's session id) that store on web server or middle tier server module
- Ex: Joomla, Wordpress, Mantis (Bug Tracking), Normal etc

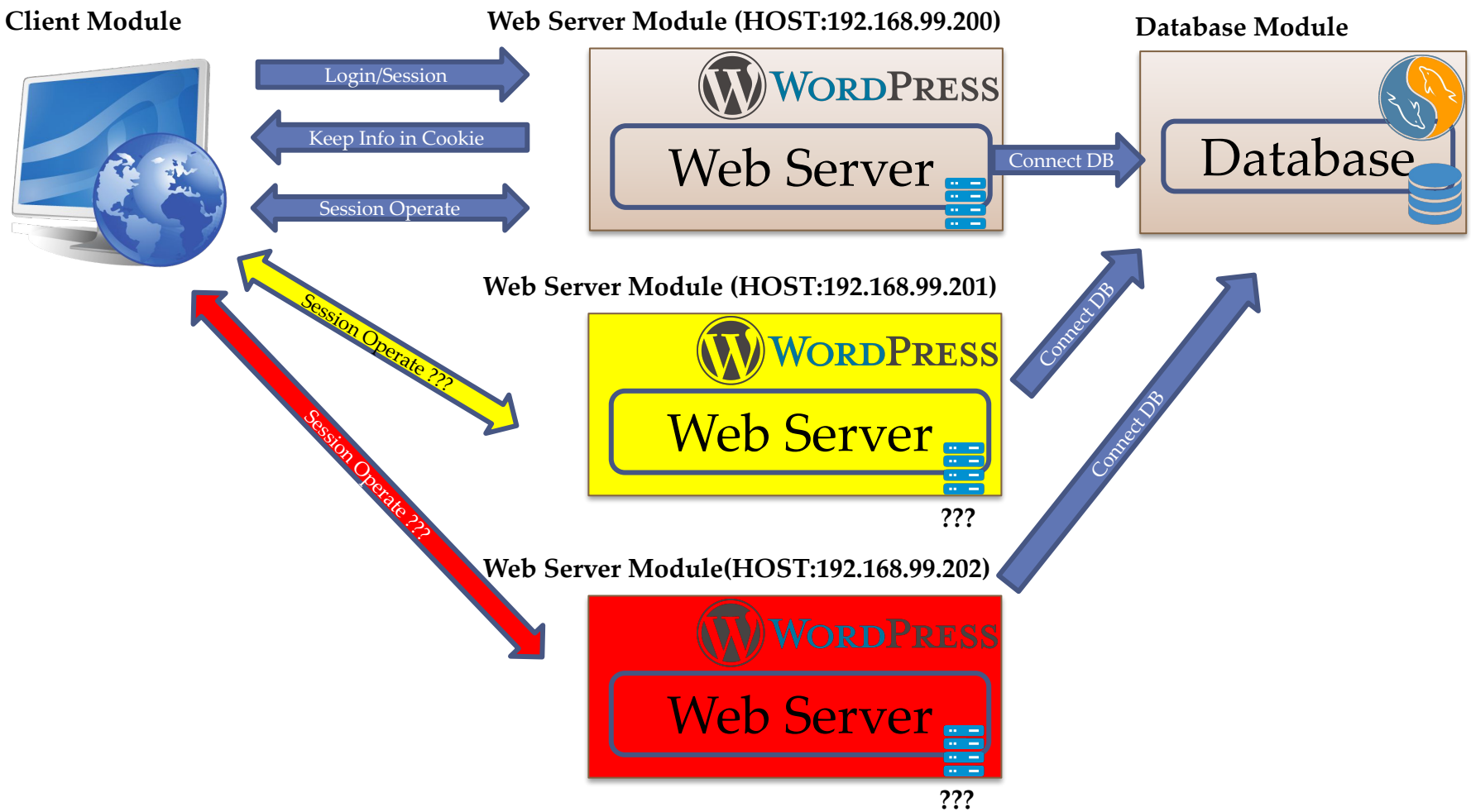


Stateful Application Deployment

- Considering
  - Original “HTTP” protocol is “stateless”
  - Stateful application need to keep session by web/app server and keep “cookies” on client for pass authentication
  - Work on memory for keep session (Fast/Easy but consume resource)
  - Many problem with native mobile app/Centralize Problem
  - Scale will effect for consideration traffic redirect to correct server (Keep state)

- Awareness
  - Container is naturally design for “stateless” application
  - All load-balance/dispatch job is not aware about “state” of application inside

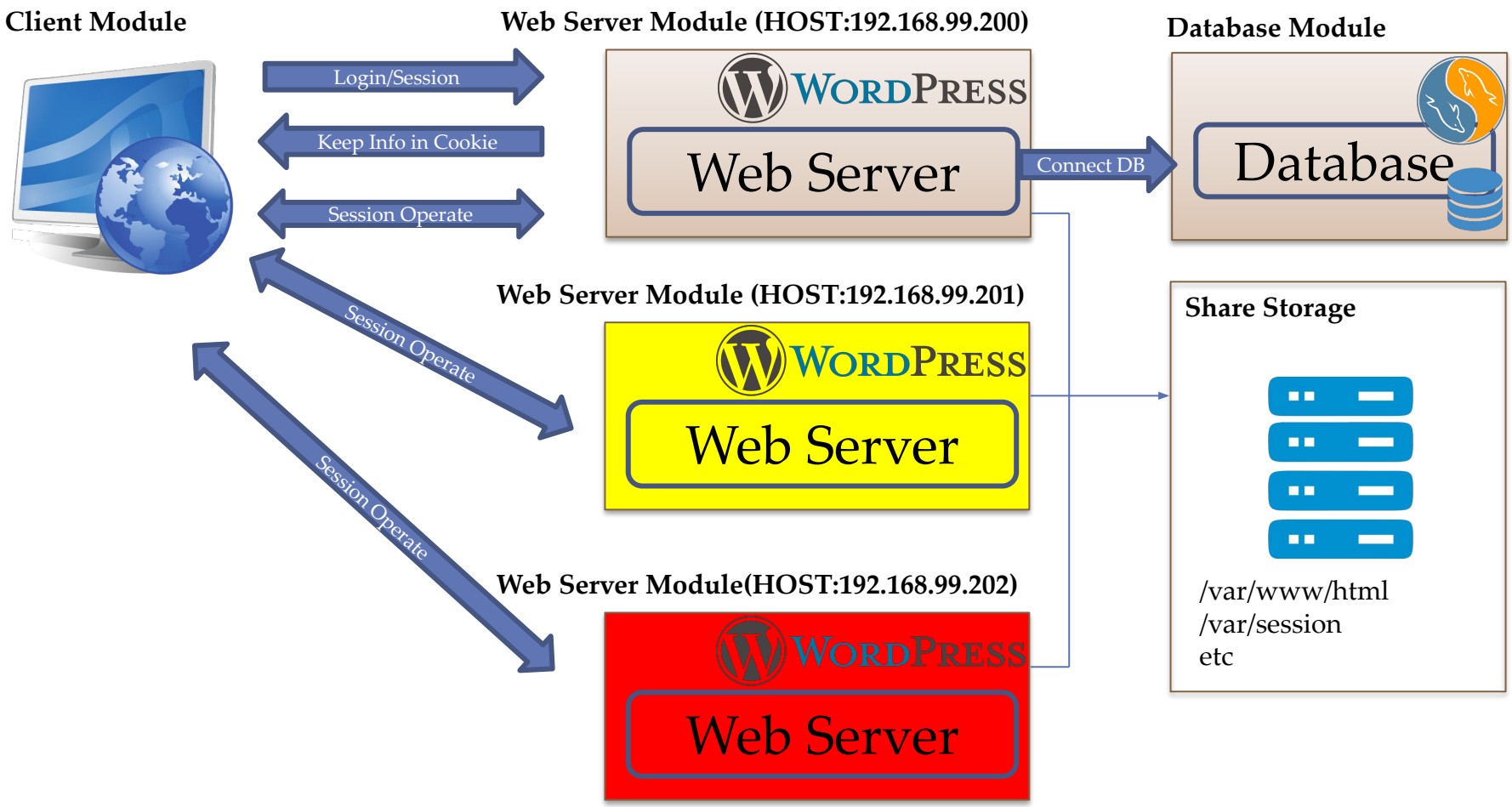
Stateful Application Deployment



Stateful Application Deployment

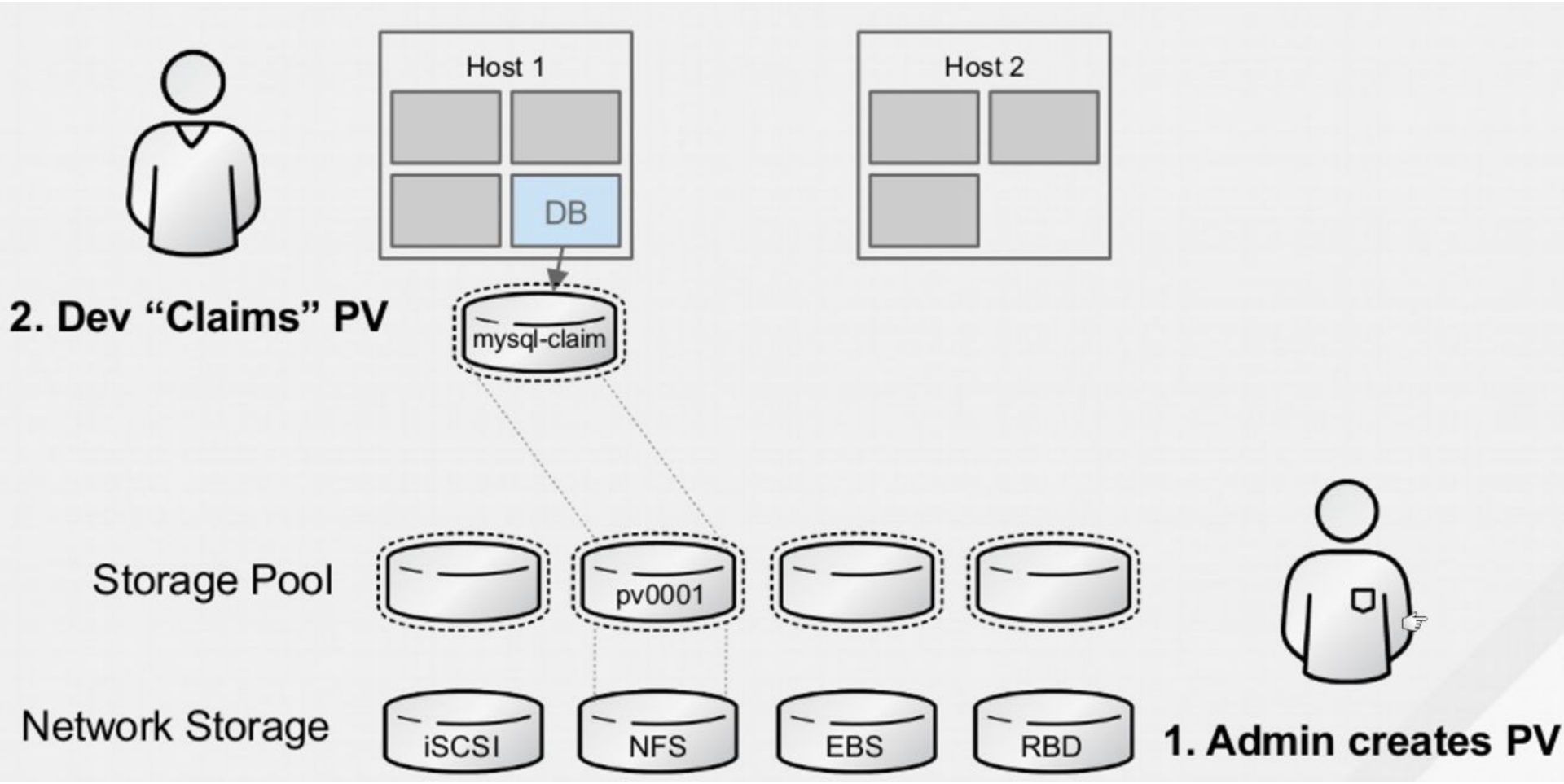
- Solution ?
  - SDS (Software-Defined Storage) for make centralize storage pool
  - Share centralize storage pool for all node
  - For Web/App Server
    - Keep application path / Session path on storage pool
    - Every server will read/write on same place
  - For Database Server
    - Many option for operate (depend on type of database)
      - Active/Active
      - Active/Hot-Passive
      - Active/Cold-Passive
      - Postgres Kubedb Tool (Beta)
    - Idea also keep data on storage pool

Stateful Application Deployment



Persistent Volume

Persistent Volume



Persistent Volume

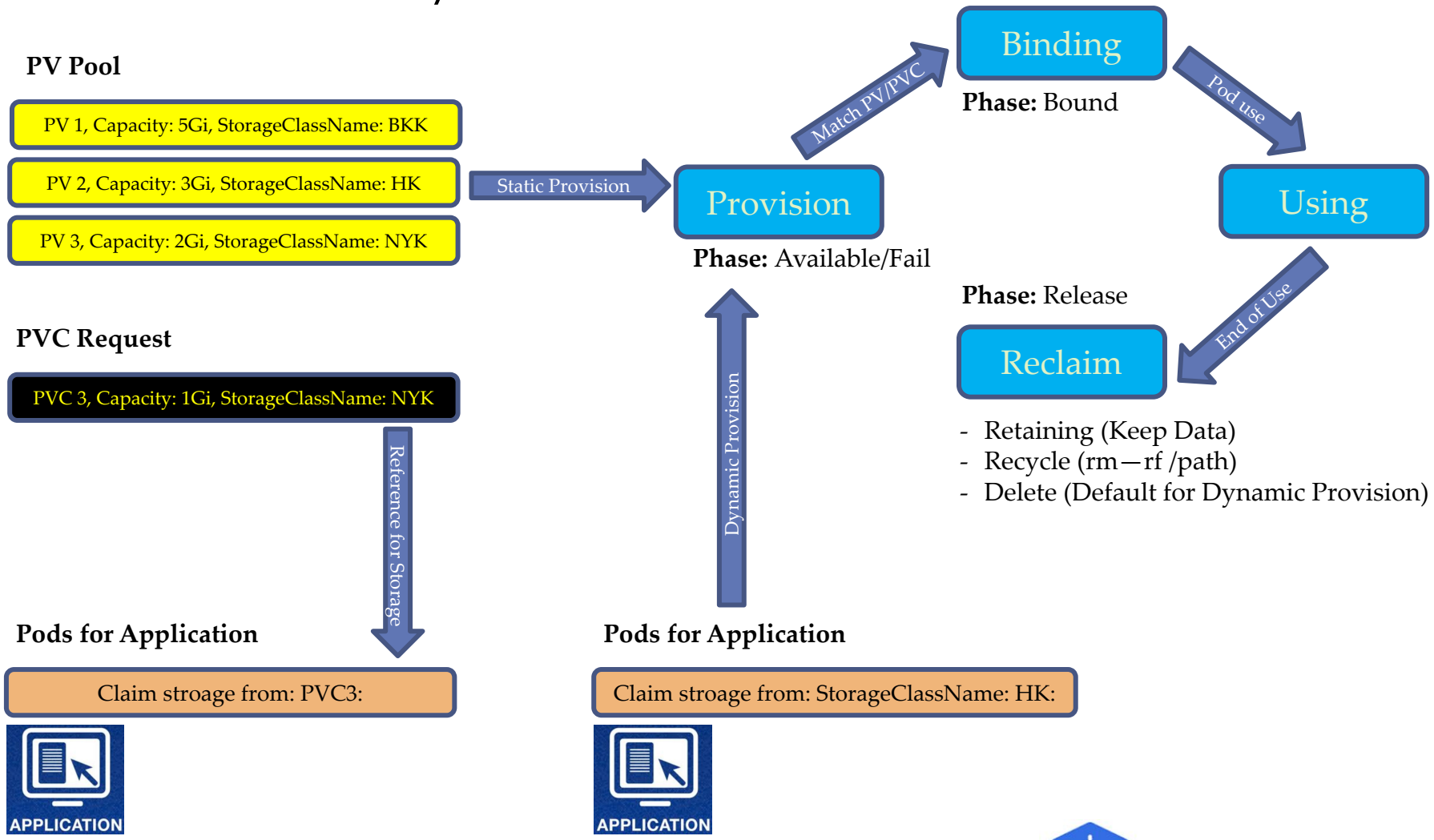
- Persistent Volume (PV)
  - Resource in cluster system
  - Act like pieces of storage (Independence from Pods)
  - Lifecycle was depended on Pods who used PV via PVC(Persistent Volume Claim)
  - Multiple type of PV as plugin support

- Persistent Volume Claim (PVC)
  - Similar Pod, That create for request storage from PV
  - PVC can specific
    - Size: Datasize for claim storage
    - Access Method:
      - ReadWriteOnce (RWO)
      - ReadOnlyMany (ROX)
      - ReadWriteMany (RWX)

- StorageClass
  - "Profiling" storage concept
  - Easy to classify storage (IOPS, Region etc)

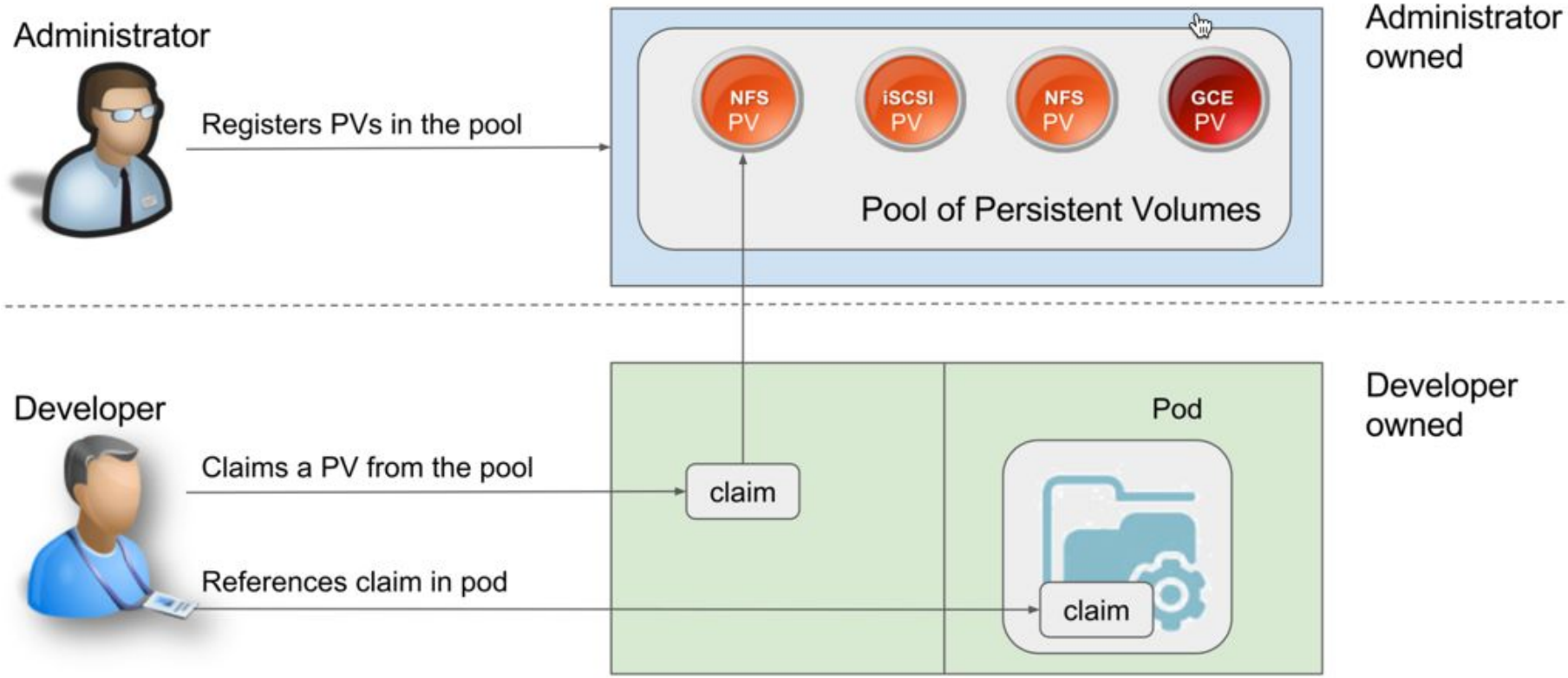
Persistent Volume

- Volume Life Cycle



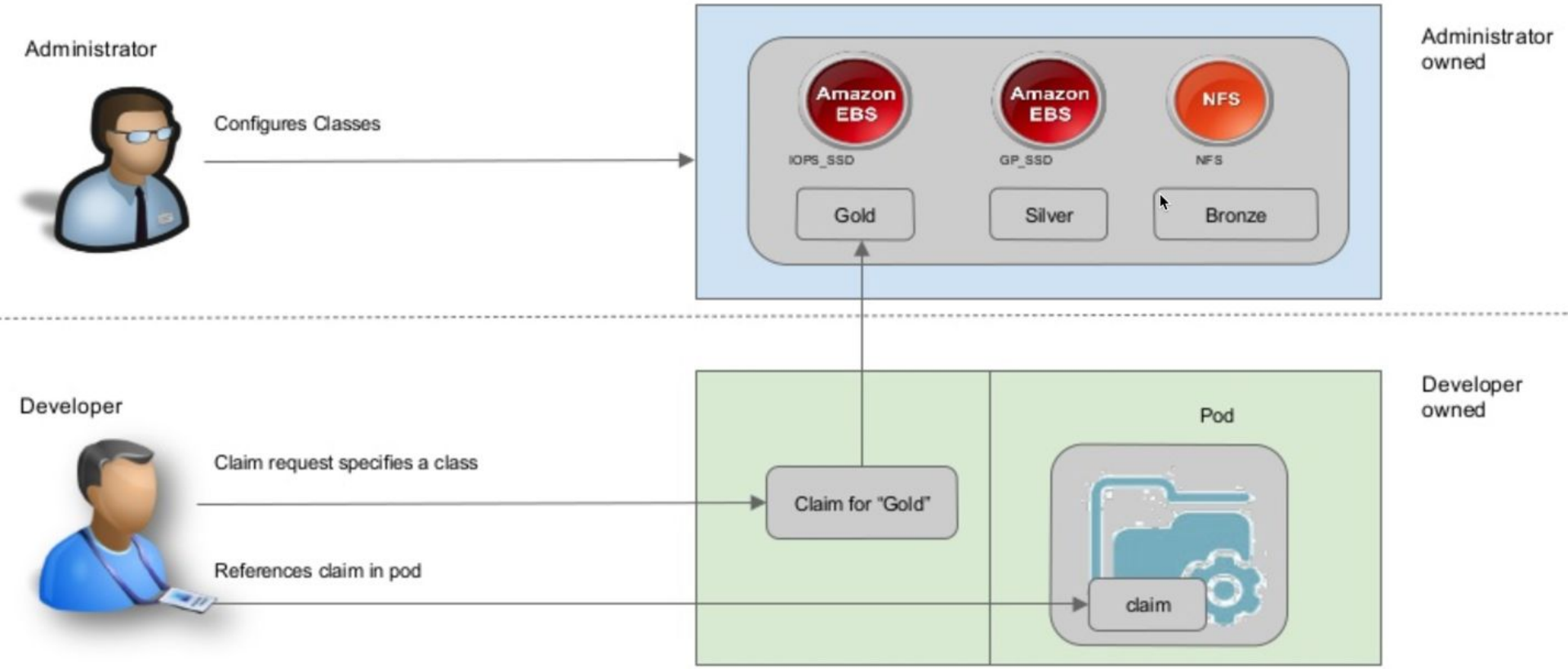
Persistent Volume

- Static Provision



Persistent Volume

- Dynamic Provision



Persistent Volume

- Type of Persistent Volume/Access Method

Volume Plugin	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
AWSElasticBlockStore	✓	-	-
AzureFile	✓	✓	✓
AzureDisk	✓	-	-
CephFS	✓	✓	✓
Cinder	✓	-	-
FC	✓	✓	-
FlexVolume	✓	✓	-
Flocker	✓	-	-
GCEPersistentDisk	✓	✓	-
Glusterfs	✓	✓	✓
HostPath	✓	-	-
iSCSI	✓	✓	-
PhotonPersistentDisk	✓	-	-
Quobyte	✓	✓	✓
NFS	✓	✓	✓
RBD	✓	✓	-
VsphereVolume	✓	-	-
PortworxVolume	✓	-	✓
ScaleIO	✓	✓	-
StorageOS	✓	-	-

Persistent Volume

Kubernetes 1.29: Single Pod Access Mode for PersistentVolumes Graduates to Stable

Monday, December 18, 2023

Author: Chris Henzie (Google)

With the release of Kubernetes v1.29, the `ReadWriteOncePod` volume access mode has graduated to general availability: it's part of Kubernetes' stable API. In this blog post, I'll take a closer look at this access mode and what it does.

What is `ReadWriteOncePod`?

`ReadWriteOncePod` is an access mode for PersistentVolumes (PVs) and PersistentVolumeClaims (PVCs) introduced in Kubernetes v1.22. This access mode enables you to restrict volume access to a single pod in the cluster, ensuring that only one pod can write to the volume at a time. This can be particularly useful for stateful workloads that require single-writer access to storage.

For more context on access modes and how `ReadWriteOncePod` works read What are access modes and why are they important? in the *Introducing Single Pod Access Mode for PersistentVolumes* article from 2021.

How can I start using `ReadWriteOncePod`?

The `ReadWriteOncePod` volume access mode is available by default in Kubernetes versions v1.27 and beyond. In Kubernetes v1.29 and later, the Kubernetes API always recognizes this access mode.

Note that `ReadWriteOncePod` is only supported for CSI volumes, and before using this feature, you will need to update the following CSI sidecars to these versions or greater:

- csi-provisioner:v3.0.0+
- csi-attacher:v3.3.0+
- csi-resizer:v1.3.0+

To start using `ReadWriteOncePod`, you need to create a PVC with the `ReadWriteOncePod` access mode:

Persistent Volume

Volume health monitoring

Kubernetes *volume health monitoring* is part of how Kubernetes implements the Container Storage Interface (CSI). Volume health monitoring feature is implemented in two components: an External Health Monitor controller, and the kubelet.

If a CSI Driver supports Volume Health Monitoring feature from the controller side, an event will be reported on the related PersistentVolumeClaim (PVC) when an abnormal volume condition is detected on a CSI volume.

The External Health Monitor controller also watches for node failure events. You can enable node failure monitoring by setting the `enable-node-watcher` flag to true. When the external health monitor detects a node failure event, the controller reports an Event will be reported on the PVC to indicate that pods using this PVC are on a failed node.

If a CSI Driver supports Volume Health Monitoring feature from the node side, an Event will be reported on every Pod using the PVC when an abnormal volume condition is detected on a CSI volume. In addition, Volume Health information is exposed as Kubelet VolumeStats metrics. A new metric kubelet_volume_stats_health_status_abnormal is added. This metric includes two labels: `namespace` and `persistentvolumeclaim`. The count is either 1 or 0. 1 indicates the volume is unhealthy, 0 indicates volume is healthy. For more information, please check KEP.

> **Note:** You need to enable the `CSIVolumeHealth` feature gate to use this feature from the node side.

Persistent Volume

Production Drivers

Name	CSI Driver Name	Compatible with CSI Version(s)	Description	Persistence (Beyond Pod Lifetime)	Supported Access Modes	Dynamic Provisioning	Other Features
Alicloud Disk	`diskplugin.csi.alibabacloud.com`	v1.0	A Container Storage Interface (CSI) Driver for Alicloud Disk	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot
Alicloud NAS	`nasplugin.csi.alibabacloud.com`	v1.0	A Container Storage Interface (CSI) Driver for Alicloud Network Attached Storage (NAS)	Persistent	Read/Write Multiple Pods	No	
Alicloud OSS	`ossplugin.csi.alibabacloud.com`	v1.0	A Container Storage Interface (CSI) Driver for Alicloud Object Storage Service (OSS)	Persistent	Read/Write Multiple Pods	No	
Alluxio	`csi.alluxio.com`	v1.0	A Container Storage Interface (CSI) Driver for Alluxio File System)	Persistent	Read/Write Multiple Pods	Yes	
ArStor CSI	`arstor.csi.huayun.io`	v1.0	A Container Storage Interface (CSI) Driver for Huayun Storage Service (ArStor)	Persistent and Ephemeral	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion, Cloning

Ref:https://kubernetes-csi.github.io/docs/drivers.html

Persistent Volume

AWS Elastic Block Storage	ebs.csi.aws.com	v0.3, v1.0	A Container Storage Interface (CSI) Driver for AWS Elastic Block Storage (EBS)	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion
AWS Elastic File System	efs.csi.aws.com	v0.3, v1.0	A Container Storage Interface (CSI) Driver for AWS Elastic File System (EFS)	Persistent	Read/Write Multiple Pods	No	
AWS FSx for Lustre	fsx.csi.aws.com	v0.3, v1.0	A Container Storage Interface (CSI) Driver for AWS FSx for Lustre (EBS)	Persistent	Read/Write Multiple Pods	Yes	
Azure Blob	blob.csi.azure.com	v1.0	A Container Storage Interface (CSI) Driver for Azure Blob storage	Persistent	Read/Write Multiple Pods	Yes	Expansion
Azure Disk	disk.csi.azure.com	v1.0	A Container Storage Interface (CSI) Driver for Azure Disk	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion, Cloning, Topology
Azure File	file.csi.azure.com	v1.0	A Container Storage Interface (CSI) Driver for Azure File	Persistent	Read/Write Multiple Pods	Yes	Expansion
BeeGFS	beegfs.csi.netapp.com	v1.3	A Container Storage Interface (CSI) Driver for the BeeGFS Parallel File System	Persistent	Read/Write Multiple Pods	Yes	
			A Container Storage				

Ref:https://kubernetes-csi.github.io/docs/drivers.html

Persistent Volume

Bigtera VirtualStor (block)	`csi.block.bigtera.com`	v0.3, v1.0.0, v1.1.0	A Container Storage Interface (CSI) Driver for Bigtera VirtualStor block storage	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion
Bigtera VirtualStor (filesystem)	`csi.fs.bigtera.com`	v0.3, v1.0.0, v1.1.0	A Container Storage Interface (CSI) Driver for Bigtera VirtualStor filesystem	Persistent	Read/Write Multiple Pods	Yes	Expansion
BizFlyCloud Block Storage	`volume.csi.bizflycloud.vn`	v1.2	A Container Storage Interface (CSI) Driver for BizFly Cloud block storage	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion
CephFS	`cephfs.csi.ceph.com`	v0.3, >=v1.0.0	A Container Storage Interface (CSI) Driver for CephFS	Persistent	Read/Write Multiple Pods	Yes	Expansion, Snapshot, Cloning
Ceph RBD	`rbd.csi.ceph.com`	v0.3, >=v1.0.0	A Container Storage Interface (CSI) Driver for Ceph RBD	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion, Topology, Cloning,In-tree plugin migration
Cisco HyperFlex CSI	`HX-CSI`	v1.2	A Container Storage Interface (CSI) Driver for Cisco HyperFlex	Persistent	Read/Write Multiple Pods	Yes	Raw Block, Expansion, Cloning
ChubaoFS	`csi.chubaofs.com`	v1.0.0	A Container Storage Interface (CSI) Driver for ChubaoFS Storage	Persistent	Read/Write Multiple Pods	Yes	

Ref:https://kubernetes-csi.github.io/docs/drivers.html

Persistent Volume

Cinder	`cinder.csi.openstack.org`	v0.3, v1.0, v1.1.0, v1.2.0, v1.3.0	A Container Storage Interface (CSI) Driver for OpenStack Cinder	Persistent and Ephemeral	Depends on the storage backend used	Yes, if storage backend supports it	Raw Block, Snapshot, Expansion, Cloning, Topology
cloudscale.ch	`csi.cloudscale.ch`	v1.0	A Container Storage Interface (CSI) Driver for the cloudscale.ch IaaS platform	Persistent	Read/Write Single Pod	Yes	Snapshot
Datatom-InfinityCSI	`csi-infiblock-plugin`	v0.3, v1.0.0, v1.1.0	A Container Storage Interface (CSI) Driver for DATATOM Infinity storage	Persistent	Read/Write Single Pod	Yes	Raw Block, Snapshot, Expansion, Topology
Datatom-InfinityCSI (filesystem)	`csi-infifs-plugin`	v0.3, v1.0.0, v1.1.0	A Container Storage Interface (CSI) Driver for DATATOM Infinity filesystem storage	Persistent	Read/Write Multiple Pods	Yes	Expansion
Datera	`dsp.csi.daterainc.io`	v1.0	A Container Storage Interface (CSI) Driver for Datera Data Services Platform (DSP)	Persistent	Read/Write Single Pod	Yes	Snapshot
DDN EXAScaler	`exa.csi.ddn.com`	v1.0, v1.1	A Container Storage Interface (CSI) Driver for DDN EXAScaler filesystems	Persistent	Read/Write Multiple Pods	Yes	Expansion

Ref:https://kubernetes-csi.github.io/docs/drivers.html

Persistent Volume

aws

Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

Blog Home Category Edition Follow

Containers

Using EBS Snapshots for persistent storage with your EKS cluster

by Leo Drakopoulos and Marco Ballerini | on 17 DEC 2020 | in Amazon Elastic Block Store (Amazon EBS), Amazon Elastic Kubernetes Service, Containers, Technical How-To | Permalink | Comments | Share

Originally, containers were a great fit for stateless applications. However, for many use cases there is a need for persistent storage, without which stateful workloads are not possible. Kubernetes first introduced support for stateful workloads with in-tree volume plugins, meaning that the plugin code was part of the core Kubernetes code and shipped with the Kubernetes binaries. That proved quite challenging for vendors who wanted to add Kubernetes support to their storage platform, as these features needed to wait for the standard Kubernetes release cycle.

That led to the development of Container Storage Interface (CSI), a standard for exposing arbitrary block and file storage systems to containerised workloads on container orchestration systems like Kubernetes.

Amazon Elastic Block Stock (Amazon EBS) is a cloud block storage service that provides direct access from an EC2 instance to a dedicated storage volume.

Support for EBS initially launched as mentioned above as an in-tree volume plugin in Kubernetes.

When the CSI specification was published, we started developing a compatible driver for Amazon EBS. We made the EBS CSI driver available and open source on GitHub. You can find the list of features supported from the EBS CSI driver here.

In this post, we are going to focus on a specific EBS CSI driver feature: Kubernetes Volume Snapshots. Kubernetes Volume Snapshots lets you create a copy of your EBS volume at a specific point in time. You can use this copy to bring a volume back to a prior state or to provision a new volume.

From version 1.17 and newer, you can provision and attach EBS volume snapshots to your pods with the following components:

- VolumeSnapshotClass
- VolumeSnapshot

You can verify the creation of a new EBS Snapshot by navigating to the EC2 and then Volumes tab in the AWS Management console. You should see something like the following:

Create Snapshot | Actions

Owned By Me | Filter by tags and attributes or search by keyword | 1 to 2 of 2

Name	Snapshot ID	Size	Description	Status	Started
	snap-0ffb0d77d59ef98f3	4 GiB	Created by AWS EBS CSI driver for volume vol-05fd72d6946bca299	completed	October 16, 2020 at 5:28:03 .

You can also verify the Volume Snapshot creation by executing the following command:

```
kubectl get volumesnapshot
```

You should see something similar to the below:

NAME	READYTOUSE	SOURCEPVC	SOURCESNAPSHOTCONTENT	RESTORESIZE	SNAPSHOTCLASS	SNAPSHOTCONTENT	CREATIONTIME	AGE
test-snapshot	true	ebs-claim		4Gi	test-snapclass	snapcontent-650e4531-89d0-407e-8fa9-25f81cc94805	2d3h	2d3h

Step 5: Create a persistent volume by restoring the Volume Snapshot

First, you need to create a Persistent Volume Claim (PVC) using the Volume Snapshot. Note that the claim this time is 20 GB in size.

Create a YAML file and Copy the YAML definition below in the **volume-from-snap.yaml file.**

YAML

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-restore
  namespace: default
spec:
  storageClassName: ebs-sc
  dataSource:
    name: test-snapshot
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
```

Ref:https://aws.amazon.com/th/blogs/containers/using-ebs-snapshots-for-persistent-storage-with-your-eks-cluster/

kubernetes by Google

Persistent Volume

- Type of Persistent Volume/Access Method

README.md

build unknown | coverage 80% | go report A+

Amazon Elastic Block Store (EBS) CSI driver

Overview

The Amazon Elastic Block Store Container Storage Interface (CSI) Driver provides a CSI interface used by Con
Orchestrators to manage the lifecycle of Amazon EBS volumes.

CSI Specification Compatibility Matrix

AWS EBS CSI Driver \ CSI Version	v0.3.0	v1.0.0	v1.1.0
master branch	no	no	yes
v0.4.0-v1.5.x	no	no	yes
v0.2.0-v0.3.0	no	yes	no
v0.1.0	yes	no	no

README.md

Deploy driver

Please see the compatibility matrix above before you deploy the driver

To deploy the CSI driver:

```
kubectl apply -k "github.com/kubernetes-sigs/aws-ebs-csi-driver/deploy/kubernetes/overlays/stable/?r
```

Verify driver is running:

```
kubectl get pods -n kube-system
```

Alternatively, you could also install the driver using helm:

Add the aws-ebs-csi-driver Helm repository:

```
helm repo add aws-ebs-csi-driver https://kubernetes-sigs.github.io/aws-ebs-csi-driver
helm repo update
```

Then install a release of the driver using the chart

```
helm upgrade --install aws-ebs-csi-driver \
    --namespace kube-system \
    aws-ebs-csi-driver/aws-ebs-csi-driver
```

Ref: https://github.com/kubernetes-sigs/aws-ebs-csi-driver

Persistent Volume

- Static Provision
- Persistent Volume

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-share-pv
  labels:
    name: nfs-share-pv
    owner: Praparn_L
    version: "1.0"
    module: PV
    environment: development
spec:
  capacity:
    storage: 1Gi
  storageClassName: ""
  accessModes:
    - ReadWriteMany
  nfs:
    server: 192.168.99.200
    path: "/var/nfsshare"
```

Persistent Volume Claims

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nfs-share-pvc
  labels:
    name: nfs-share-pvc
    owner: Praparn_L
    version: "1.0"
    module: PVC
    environment: development
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: ""
  resources:
    requests:
      storage: 500Mi
  selector:
    matchLabels:
      name: nfs-share-pv
      owner: Praparn_L
      version: "1.0"
      module: PV
      environment: development
```

Persistent Volume

- Dynamic Provision
- Storage Class

```yaml
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: ebs-sc
provisioner: ebs.csi.aws.com
parameters:
  type: gp3
  fsType: ext4
```

```yaml
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: ebs-sc
provisioner: ebs.csi.aws.com
parameters:
  type: gp3
  fsType: ext4
  encrypted: "true"
```

Persistent Volume Claims

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: aws-ebs-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: ebs-sc
```

Persistent Volume

- CSI Storage Resizing Authenticated (New on 1.29)

Secret

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: storagecsi-storage
  namespace: default
data:
  stringData:
    username: admin
    password: nevergiveup
```

StorageClass

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-blockstorage-sc
parameters:
  csi.storage.k8s.io/node-expand-secret-name: storagecsi-storage
  csi.storage.k8s.io/node-expand-secret-namespace: default
provisioner: blockstorage.cloudprovider.someprovider
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
```

Persistent Volume

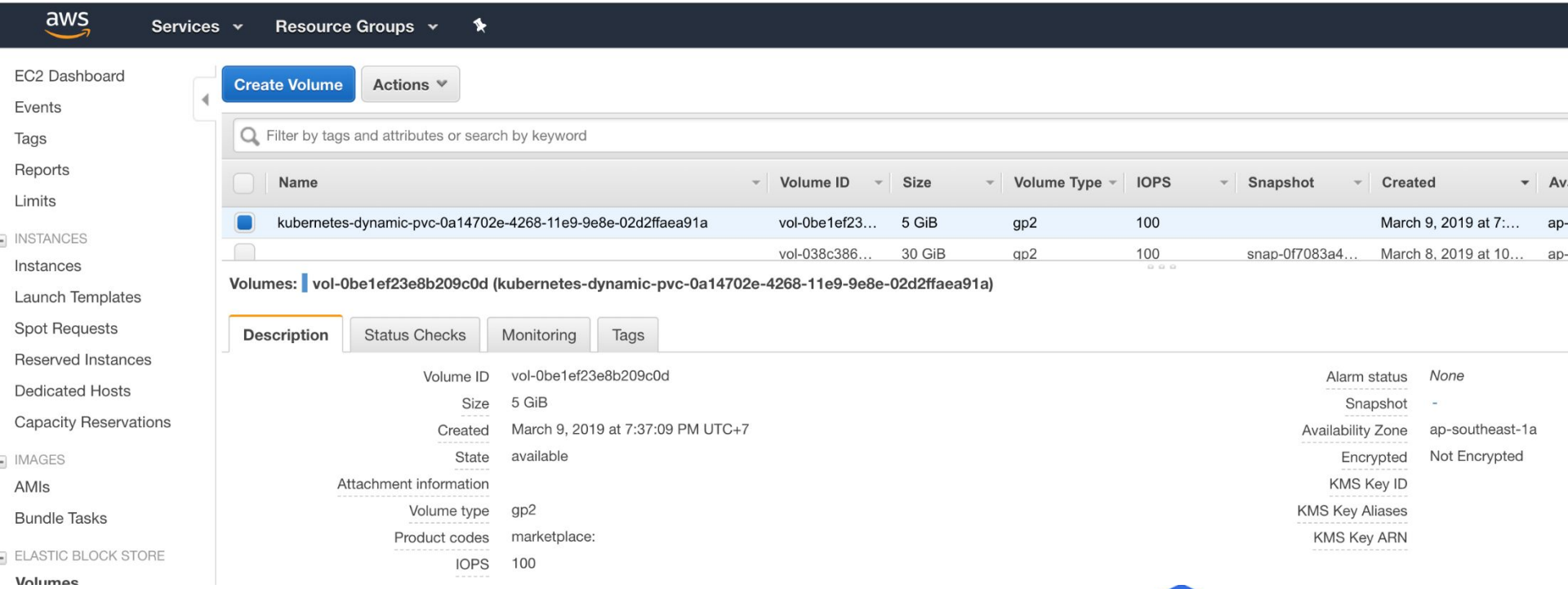
- Dynamic Provision
  - Test create SC/PVC

```
ubuntu@ip-10-0-1-229:~$ kubectl apply -f ~/kubernetes_201904/WorkShop_2.7_Persistent_Storage/aws_sc.yml
storageclass.storage.k8s.io/aws-ebs created
ubuntu@ip-10-0-1-229:~$ kubectl describe -f ~/kubernetes_201904/WorkShop_2.7_Persistent_Storage/aws_sc.yml
Name:            aws-ebs
IsDefaultClass:  No
Annotations:     kubectl.kubernetes.io/last-applied-configuration={"apiVersion":"storage.k8s.io/v1","kind":"StorageClass","metadata":{"annotations":{},"name":"aws-ebs"},"par
ameters":{"fsType":"ext4","type":"gp2"},"provisioner":"kubernetes.io/aws-ebs"}

Provisioner:           kubernetes.io/aws-ebs
Parameters:            fsType=ext4,type=gp2
AllowVolumeExpansion:  <unset>
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     Immediate
Events:                <none>
ubuntu@ip-10-0-1-229:~$ kubectl describe sc/aws-ebs
Name:            aws-ebs
IsDefaultClass:  No
Annotations:     kubectl.kubernetes.io/last-applied-configuration={"apiVersion":"storage.k8s.io/v1","kind":"StorageClass","metadata":{"annotations":{},"name":"aws-ebs"},"par
ameters":{"fsType":"ext4","type":"gp2"},"provisioner":"kubernetes.io/aws-ebs"}

Provisioner:           kubernetes.io/aws-ebs
Parameters:            fsType=ext4,type=gp2
AllowVolumeExpansion:  <unset>
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     Immediate
Events:                <none>
ubuntu@ip-10-0-1-229:~$ kubectl apply -f ~/kubernetes_201904/WorkShop_2.7_Persistent_Storage/aws_pvc.yml
persistentvolumeclaim/aws-ebs-pvc created
ubuntu@ip-10-0-1-229:~$ kubectl get pvc
NAME          STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
aws-ebs-pvc   Pending                                     aws-ebs        4s
ubuntu@ip-10-0-1-229:~$ kubectl get pvc
NAME          STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
aws-ebs-pvc   Bound    pvc-0a14702e-4268-11e9-9e8e-02d2ffaea91a   5Gi        RWO            aws-ebs        18s
```

Persistent Volume

- Dynamic Provision
  - Test create SC/PVC

```
ubuntu@ip-10-0-1-229:~$ kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS   REASON   AGE
pvc-0a14702e-4268-11e9-9e8e-02d2ffaea91a   5Gi        RWO            Delete           Bound    default/aws-ebs-pvc    aws-ebs                 26m
ubuntu@ip-10-0-1-229:~$ kubectl get pvc
NAME          STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
aws-ebs-pvc   Bound    pvc-0a14702e-4268-11e9-9e8e-02d2ffaea91a   5Gi        RWO            aws-ebs        26m
ubuntu@ip-10-0-1-229:~$
```



Persistent Volume

- Dynamic Provision

kubernetes

Search

+ CREATE

Cluster > Storage Classes > aws-ebs

EDIT DELETE

Cluster

Namespaces

Nodes

Persistent Volumes

Roles

Storage Classes

Namespace

default

Overview

Workloads

Cron Jobs

Details

Name:	aws-ebs
Annotations:	kubectl.kubernetes.io/last-applied-configuration
Creation Time:	2019-03-09T12:34 UTC
Labels:	-
Provisioner:	kubernetes.io/aws-ebs
Parameters:	fsType: ext4 type: gp2

Persistent Volumes

Name	Capacity	Access Modes	Reclaim Policy	Status	Claim	Storage Class	Reason	Age
pvc-0a14702e-4...	5Gi	ReadWriteOnce	Delete	Bound	default/aws-ebs-...	aws-ebs	-	29 minutes

kubernetes by Google

Persistent Volume

- Dynamic Provision

kubernetes | Search | + CREATE

Cluster > Persistent Volumes

Cluster
- Namespaces
- Nodes
- Persistent Volumes
- Roles

Persistent Volumes

Name	Capacity	Access Modes	Reclaim Policy	Status	Claim	Storage Class	Reason	Age
pvc-0a14702e-4...	5Gi	ReadWriteOnce	Delete	Bound	default/aws-ebs-...	aws-ebs	-	30 minutes

kubernetes | Search | + CREATE

Config and storage > Persistent Volume Claims

default

Overview

Workloads
- Cron Jobs

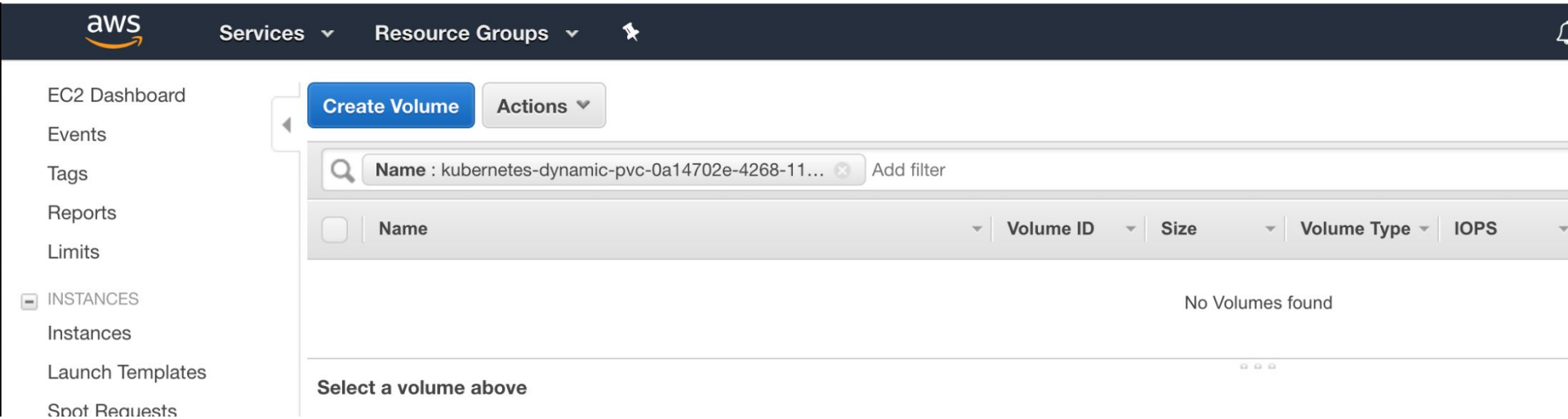
Persistent Volume Claims

Name	Status	Volume	Capacity	Access Modes	Storage Class	Age
aws-ebs-pvc	Bound	pvc-0a14702e-4268-11e9-9e8e-02d2ffaea91a	5Gi	ReadWriteOnce	-	30 minutes

Persistent Volume

- Dynamic Provision

```
ubuntu@ip-10-0-1-229:~$ kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS   REASON   AGE
pvc-0a14702e-4268-11e9-9e8e-02d2ffaea91a   5Gi        RWO            Delete           Bound    default/aws-ebs-pvc    aws-ebs                 37m
ubuntu@ip-10-0-1-229:~$ kubectl get pvc
NAME          STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
aws-ebs-pvc   Bound    pvc-0a14702e-4268-11e9-9e8e-02d2ffaea91a   5Gi        RWO            aws-ebs        37m
ubuntu@ip-10-0-1-229:~$ kubectl delete -f ~/kubernetes_201904/WorkShop_2.7_Persistent_Storage/aws_pvc.yml
persistentvolumeclaim "aws-ebs-pvc" deleted
ubuntu@ip-10-0-1-229:~$ kubectl get pv
No resources found.
ubuntu@ip-10-0-1-229:~$ kubectl get pvc
No resources found.
ubuntu@ip-10-0-1-229:~$
```



Persistent Volume

- Deployment reference

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  selector:
    matchLabels:
      name: web
      owner: "Praparn_L"
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - containerPort: 5000
            protocol: TCP
          volumeMounts:
          - name: aws-share-pvc
            mountPath: /usr/src/app
      initContainers:
        - name: download
          image: labdocker/alpineweb:latest
          volumeMounts:
          - name: aws-share-pvc
            mountPath: /source
          command:
          - wget
          - "-O"
          - "/source/mainlite.py"
          - https://raw.githubusercontent.com/praparn/kubernetes_201904/master/WorkShop_2.7_Persistent_Storage/mainlite.py
      volumes:
      - name: aws-share-pvc
        persistentVolumeClaim:
          claimName: aws-ebs-pvc
```

Run iniContainers Process

Reference for mount disk volume from “PVC”

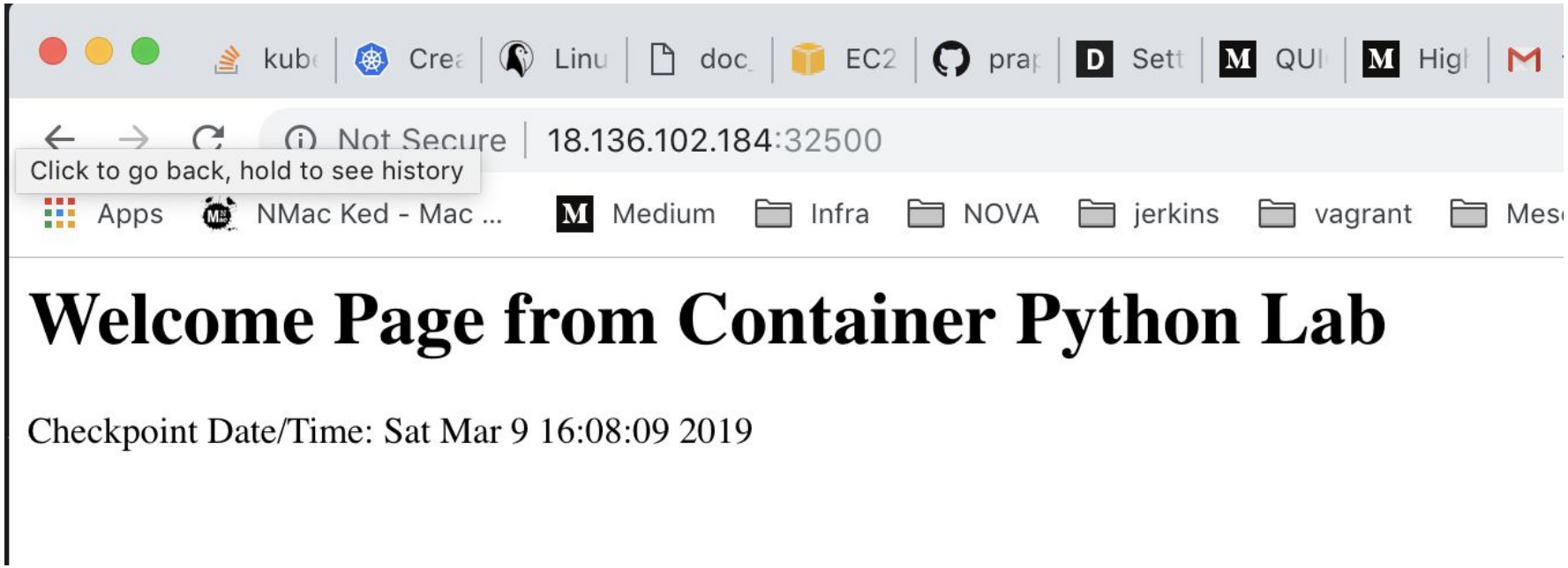
Persistent Volume

- Dynamic Provision on Deployment

```
ubuntu@ip-10-0-1-229:~$ kubectl create -f https://raw.githubusercontent.com/praparn/kubernetes_201904/master/WorkShop_2.7_Persistent_Storage/aws_webtest_deployall.yml
deployment.apps/webtest created
ubuntu@ip-10-0-1-229:~$ kubectl create -f https://raw.githubusercontent.com/praparn/kubernetes_201904/master/WorkShop_2.7_Persistent_Storage/aws_webtest_svc.yml
service/webtest created
ubuntu@ip-10-0-1-229:~$ kubectl get deployment
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
webtest   1/1     1            1           13s
ubuntu@ip-10-0-1-229:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
webtest-6f4bdd5f6b-kj4jn   1/1     Running   0          18s
ubuntu@ip-10-0-1-229:~$ kubectl describe pods/webtest-6f4bdd5f6b-kj4jn
Name:               webtest-6f4bdd5f6b-kj4jn
Namespace:          default
Priority:           0
PriorityClassName:  <none>
Node:               ip-10-0-1-165.ap-southeast-1.compute.internal/10.0.1.165
Start Time:         Sat, 09 Mar 2019 16:05:40 +0000
Labels:             environment=development
                    module=WebServer
                    name=web
                    owner=Praparn_L
                    pod-template-hash=6f4bdd5f6b
                    version=1.0
Annotations:        cni.projectcalico.org/podIP: 192.168.1.40/32
Status:             Running
IP:                 192.168.1.40
Controlled By:      ReplicaSet/webtest-6f4bdd5f6b
```

```
Events:
  Type    Reason                  Age   From                                                   Message
  ----    ------                  ----  ----                                                   -------
  Normal  Scheduled               28s   default-scheduler                                      Successfully assigned default/webtest-6f4bdd5f6b-kj4jn to ip-10-0-1-165.ap-southeast-1.compute
.internal
  Normal  SuccessfulAttachVolume  27s   attachdetach-controller                                AttachVolume.Attach succeeded for volume "pvc-e7988ede-4284-11e9-9e8e-02d2ffaea91a"
  Normal  Pulling                 23s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal pulling image "labdocker/alpineweb:latest"
  Normal  Pulled                  19s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal Successfully pulled image "labdocker/alpineweb:latest"
  Normal  Created                 19s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal Created container
  Normal  Started                 19s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal Started container
  Normal  Pulled                  18s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal Container image "labdocker/cluster:webservicelite_v1" already present on machine
  Normal  Created                 18s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal Created container
  Normal  Started                 18s   kubelet, ip-10-0-1-165.ap-southeast-1.compute.internal Started container
ubuntu@ip-10-0-1-229:~$
```

Persistent Volume



Persistent Volume

```
...-0-1-77: ~ — -bash    ...orker-1: ~ — -bash    ...-219-46: ~ — -bash
ubuntu@ip-10-0-1-229:~$ kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
webtest-6f4bdd5f6b-kj4jn    1/1     Running   0          2m51s
ubuntu@ip-10-0-1-229:~$ kubectl exec -it webtest-6f4bdd5f6b-kj4jn sh
/usr/src/app # vi mainlite.py
/usr/src/app #
```

```
from flask import Flask
import os
import time
app = Flask(__name__)


@app.route('/')
def hello():
    return '<H1>TEST EDIT Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: ' + time.strftime("%c") +'\n'

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)
~
~
```

Not Secure | 18.136.102.184:32500

Apps NMac Ked - Mac ... Medium Infra NOVA jerkins vagrant Mesos Vue do

TEST EDIT Welcome Page from Container Python Lab

Checkpoint Date/Time: Sat Mar 9 16:10:42 2019

Persistent Volume

- Create Deployment

kubernetes | Search | + CREATE

Workloads > Pods > webtest-6f4bdd5f6b-kj4jn | EXEC | LOGS | EDIT | DELETE

Cluster
- Namespaces
- Nodes
- Persistent Volumes
- Roles
- Storage Classes

Namespace: default

Overview

Workloads
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

Discovery and Load Balancing
- Ingresses
- Services

Config and Storage
- Config Maps

Details

Name:	webtest-6f4bdd5f6b-kj4jn
Namespace:	default
Labels:	environment: development, module: WebServer, name: web, owner: Praparn_L, pod-template-hash: 6f4bdd5f6b, show all
Annotations:	cni.projectcalico.org/podIP: 192.168.1.40/32
Creation Time:	2019-03-09T16:05 UTC
Status:	Running
QoS Class:	BestEffort

Network

Node:	ip-10-0-1-165.ap-southeast-1.compute.internal
IP:	192.168.1.40

Containers

webtest

Image:	labdocker/cluster:webservicelite_v1
Environment variables:	-
Commands:	-
Args:	-

Init Containers

download

Image:	labdocker/alpineweb:latest
Environment variables:	-
Commands:	wget -O /source/mainlite.py https://raw.githubusercontent.com/praparn/kubernetes_201904/master/WorkShop_2.7_Persistent_Storage/mainlite.py
Args:	-

Persistent Volume

- Create Deployment

kubernetes | Search | + CREATE

Workloads > Pods > webtest-6f4bdd5f6b-kj4jn | EXEC | LOGS | EDIT | DELETE

Cluster
- Namespaces
- Nodes
- Persistent Volumes
- Roles
- Storage Classes

Namespace: default

Overview

Workloads
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

Discovery and Load Balancing
- Ingresses
- Services

Config and Storage
- Config Maps
- Persistent Volume Claims
- Secrets

webtest-6f4bdd5f6b	replicaset	name: web owner: Praparn_L pod-template-hash: 6f4bdd5f6b show all	1 / 1	7 minutes	labdocker/cluster:webservicelit...

Events

Message	Source	Sub-object	Count	First seen	Last seen
Successfully assigned default/webtest-6f4bdd5f6b-kj4jn to ip-10-0-1-165.ap-southeast-1.compute.internal	default-scheduler	-	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
AttachVolume.Attach succeeded for volume "pvc-e7988ede-4284-11e9-9e8e-02d2ffaea91a"	attachdetach-controller	-	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
pulling image "labdocker/alpineweb:latest"	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Successfully pulled image "labdocker/alpineweb:latest"	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Created container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Started container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Container image "labdocker/cluster:webservicelite_v1" already present on machine	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.containers{webtest}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Created container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.containers{webtest}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Started container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.containers{webtest}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC

Persistent Volume Claims

Name	Status	Volume	Capacity	Access Modes	Storage Class	Age
aws-ebs-pvc	Bound	pvc-e7988ede-4284-11e9-9e8e-02d2ffaea91a	5Gi	ReadWriteOnce	-	9 minutes

Workshop: Persistence Storage

kubernetes | Search | + CREATE

Workloads > Pods > webtest-6f4bdd5f6b-kj4jn | EXEC | LOGS | EDIT | DELETE

Cluster
- Namespaces
- Nodes
- Persistent Volumes
- Roles
- Storage Classes

Namespace: default

Overview

Workloads
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

Discovery and Load Balancing
- Ingresses
- Services

Config and Storage
- Config Maps
- Persistent Volume Claims
- Secrets

webtest-6f4bdd5f6b	replicaset	name: web owner: Praparn_L pod-template-hash: 6f4bdd5f6b show all	1 / 1	7 minutes	labdocker/cluster:webservicelit...

Events

Message	Source	Sub-object	Count	First seen	Last seen
Successfully assigned default/webtest-6f4bdd5f6b-kj4jn to ip-10-0-1-165.ap-southeast-1.compute.internal	default-scheduler	-	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
AttachVolume.Attach succeeded for volume "pvc-e7988ede-4284-11e9-9e8e-02d2ffaea91a"	attachdetach-controller	-	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
pulling image "labdocker/alpineweb:latest"	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Successfully pulled image "labdocker/alpineweb:latest"	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Created container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Started container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.initContainers{download}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Container image "labdocker/cluster:webservicelite_v1" already present on machine	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.containers{webtest}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Created container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.containers{webtest}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC
Started container	kubelet ip-10-0-1-165.ap-southeast-1.compute.internal	spec.containers{webtest}	1	2019-03-09T16:05 UTC	2019-03-09T16:05 UTC

Persistent Volume Claims

Name	Status	Volume	Capacity	Access Modes	Storage Class	Age
aws-ebs-pvc	Bound	pvc-e7988ede-4284-11e9-9e8e-02d2ffaea91a	5Gi	ReadWriteOnce	-	9 minutes

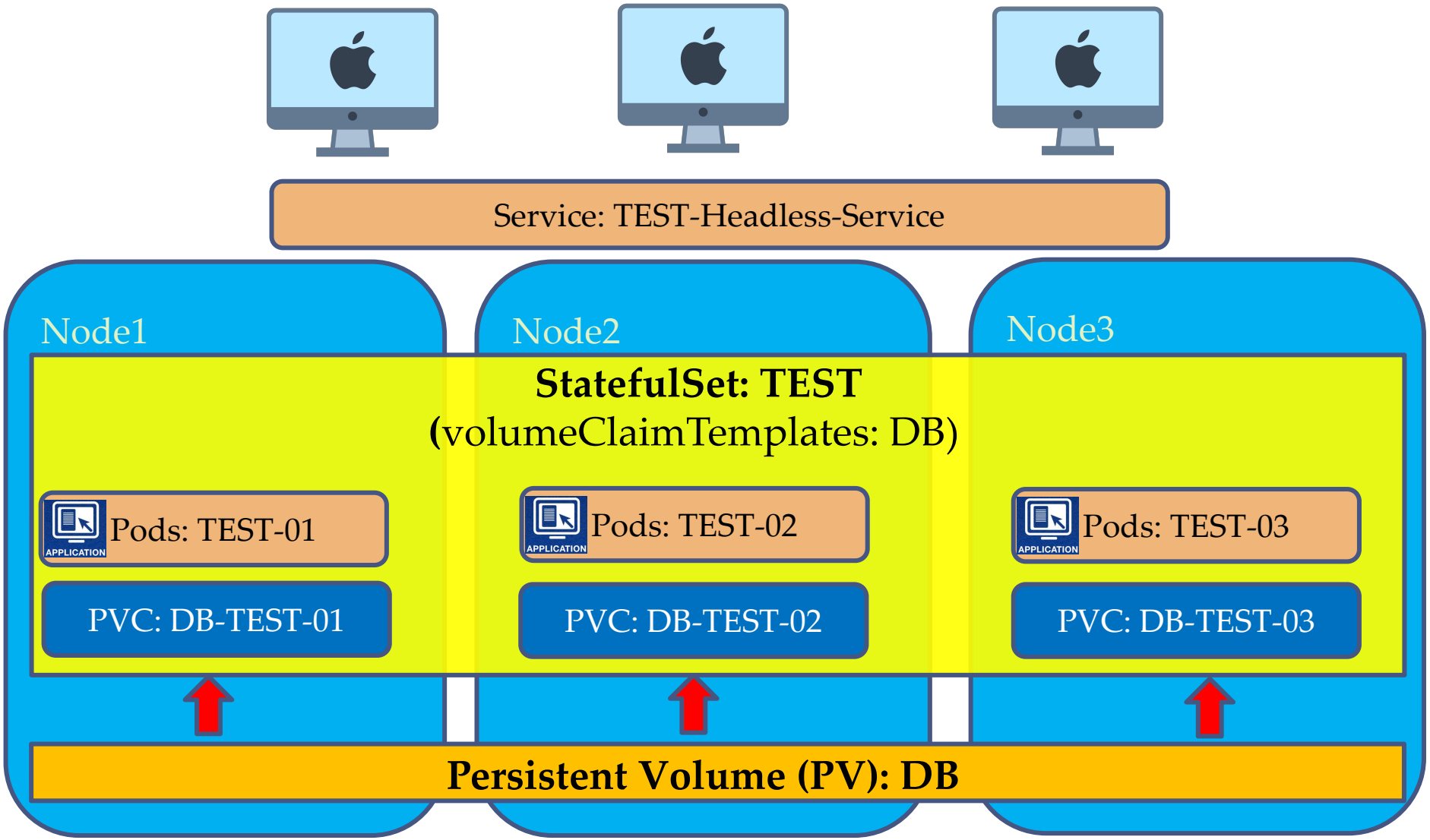
StatefulSet

StatefulSet

- What is StatefulSet?
  - Do you remember Deployment ?
  - StatefulSet similar with Deployment
  - But StatefulSet is design for applicate that need
    - Persistent Storage
    - Stable Storage/Stable Network
    - Ordered for
      - Deployment (Create)
      - Scale
      - Roll Update

- Benefit from StatefulSet
  - Sequential create/scale/update Pods
  - Each Pods got unique resource
    - Name of Pods
    - Storage on Pods (Dedicate PVC)
    - Exist on Hosts/Network

StatefulSet



Kubernetes: Production Workload Orchestration

kubernetes by Google

StatefulSet

- StatefulSet

```yaml
apiVersion: apps/v1beta2
kind: StatefulSet
metadata:
  name: TEST
spec:
  serviceName: "TEST"
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: labdocker/nginx:latest
        ports:
        - containerPort: 80
          name: web
        volumeMounts:
        - name: DB
          mountPath: /usr/share/nginx/html
  volumeClaimTemplates:
  - metadata:
      name: DB
    spec:
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 1Gi
```

Service

```yaml
apiVersion: v1
kind: Service
metadata:
  name: TEST
  labels:
    app: TEST
spec:
  ports:
  - port: 80
    name: TEST
  clusterIP: None
  selector:
    app: nginx
```

Recapture Day 2

- Fundamental of Kubernetes
  - Job and CronJob
  - Log and Monitoring
- Ingress Networking
- Security on Kubernetes
  - Network Policy
  - Volume Policy
  - Resource Usage Policy
  - Resource Consumption Policy
  - Access Control Policy
  - Security Policy
- Kubernetes in real world
  - Cluster Setup for Bare Metal
  - Orchestrator Assignment
    - nodeSelector
    - Interlude
    - Affinity
    - Taints/Tolerations
- Stateful application deployment
  - Consideration and Awareness
  - Persistent Volumes
  - StatefulSets

KhaDev PRESENT

WORKSHOP ADVANCED DOCKER

WITH

kubernetes

สอนการ deploy Dockers บน Kubernetes
จากประสบการณ์ใช้งานจริงบน Production ของ
application ระดับประเทศ

วิทยากร : คุณ PRAPARN LUNGPOONLARP
INFRASTRUCTURE ENGINEER, NETWORK ENGINEER,
SYSTEM ENGINEER

docker kubernetes KhaDev 3DS INTERACTIVE