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

DAY 1

Outline Day 1

- ***Recap Container Concept**
- Container principle
- Image, Repository & Tag, container
- CPU, Memory and I/O
- Network
- Volume

Outline Day 2

- ***Recap Container Concept**
- Inspect/Log/Commit
- Dockerfile and Build
- Compose
- Docker Security
- Registry

Outline Day 3

- 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)

Outline Day 4

- Fundamental of Kubernetes
  - Configmap and Secret
  - 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

Outline Day 5

- 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

Prerequisite

- Windows (64 bit)/ Mac / Linux (64 bit)machine (ubuntu / alpine prefer)
- 1 email address (For register “hub.docker.com”) / hub.docker.com account
- Line Notify Token (https://notify-bot.line.me)
- Tool for editor (vscode etc)
- Tool for shell (putty / terminal etc)
- Tool for transfer file (winscp / scp)
- Basic understand for linux operate
- Basic text editor skill (vim prefer) and linux structure
- Internet for download / upload image

Docker Lab Resource

- https://tinyurl.com/y9svjua2

Lab Resource

- Repository for lab

Explore Help labdocker Sign up Log In

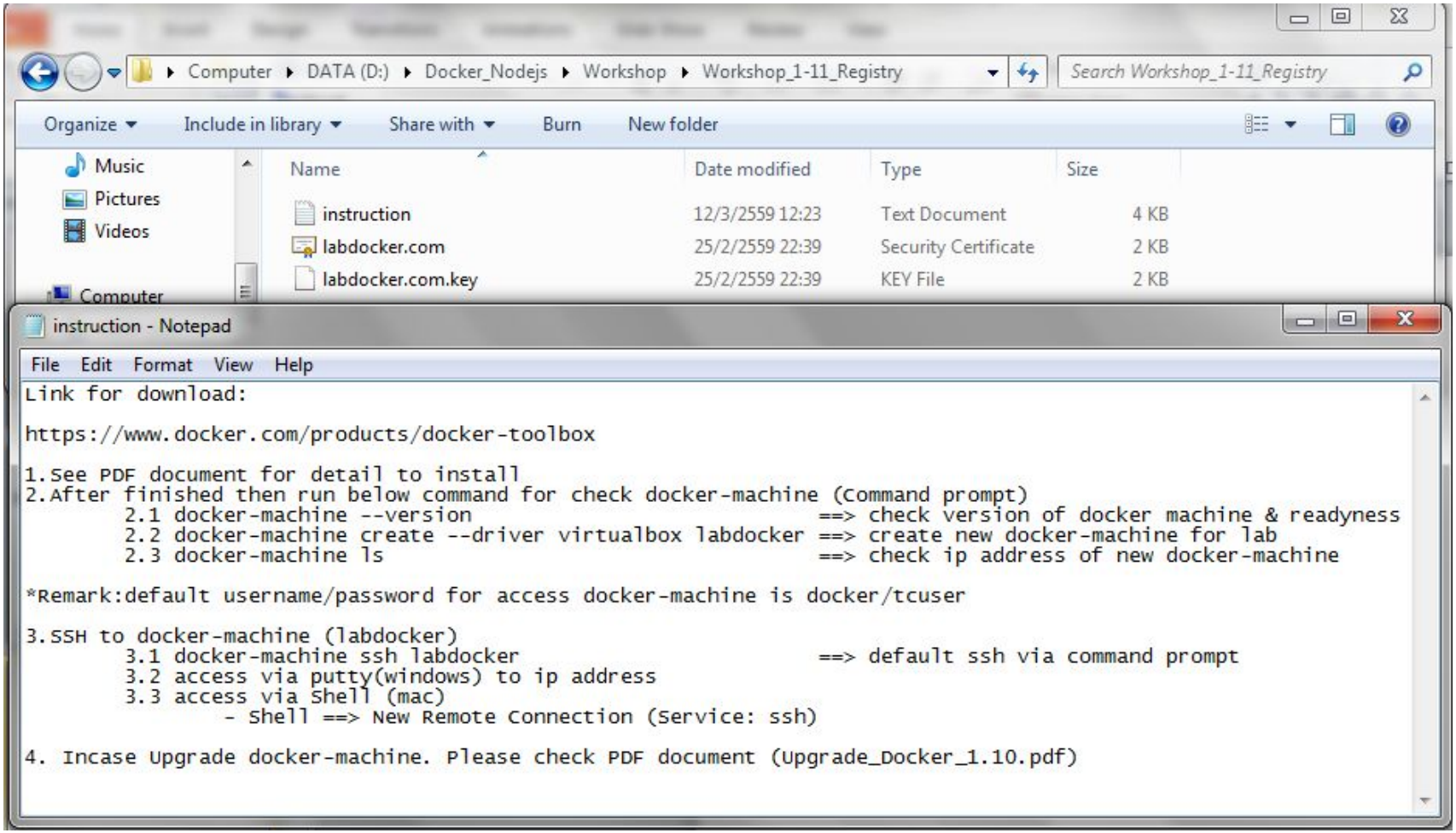
Repositories (7)

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

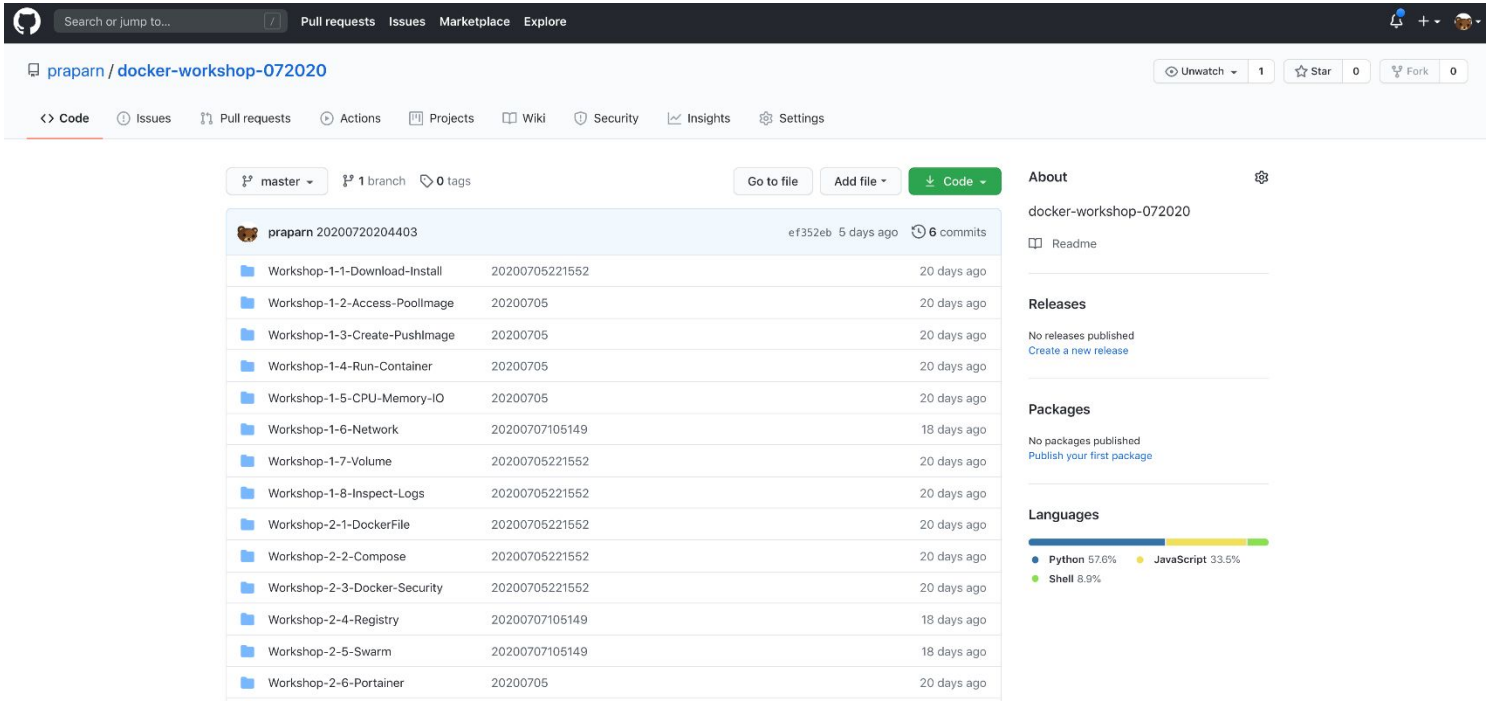
Lab Resource

- Software in lab



Lab Resource

- Download on Google Drive
  - https://tinyurl.com/feu5jhyr
- Download on GitHub
  - git clone https://github.com/praparn/docker-workshop-032022



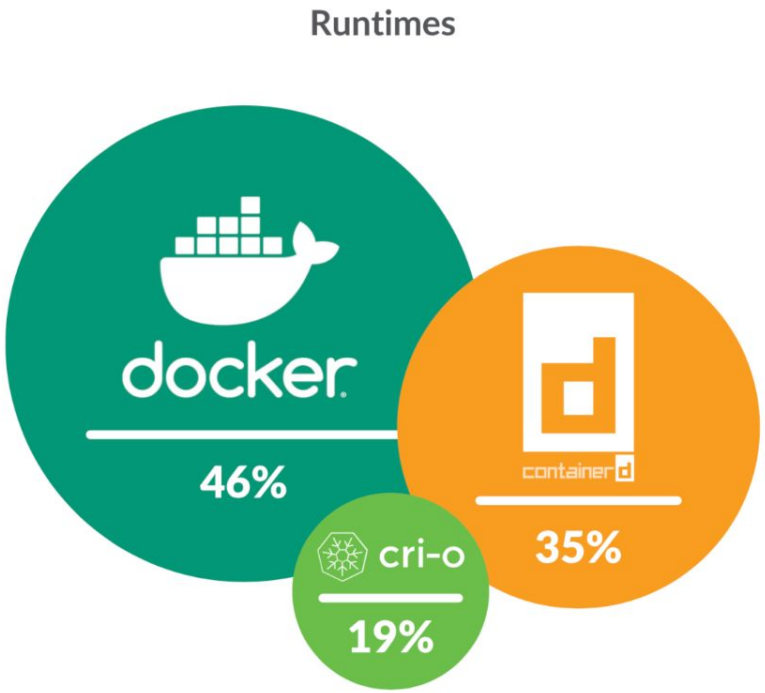
Container World Report

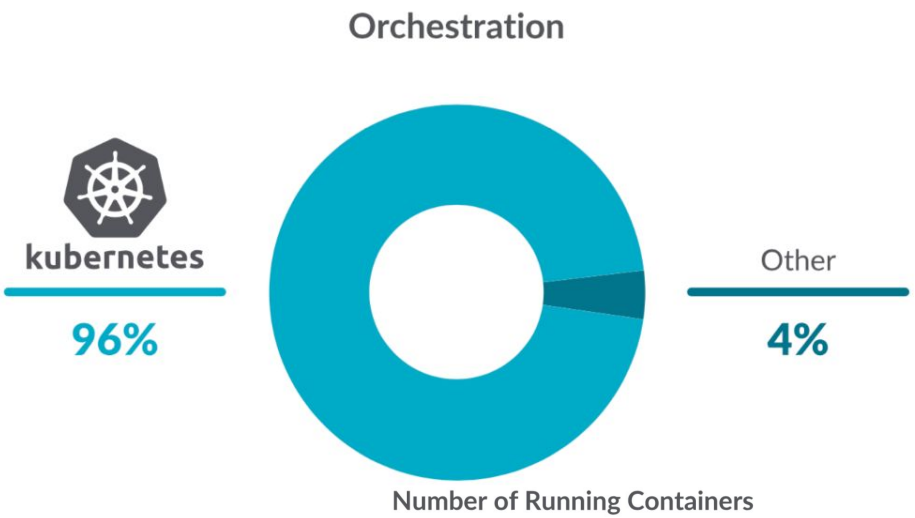
Sysdig 2022 Cloud-Native Security and Usage Report

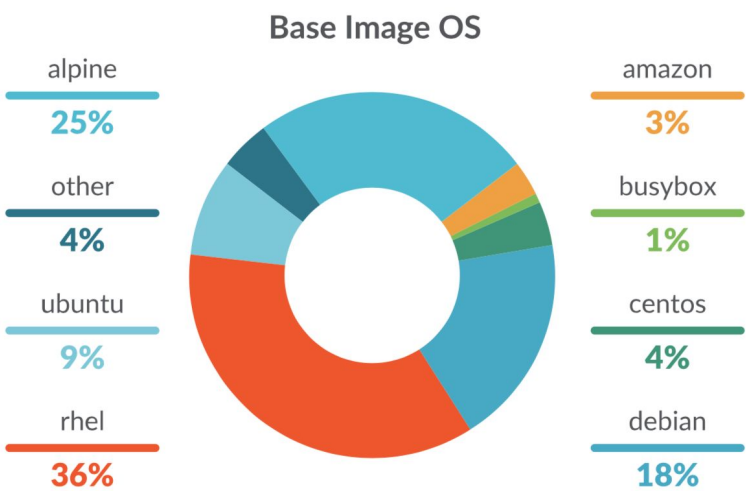
Cloud Security		Container Security		Container Usage	
88%	cloud roles are non-human	75%	containers running with "high" or "critical" vulnerabilities	34%	unused CPU resources
73%	cloud accounts have public S3 buckets	62%	detect shell in container events	51%	no memory limits
				60%	no CPU limits
73%	YoY growth in Falco downloads	76%	containers running as root	83%	custom metrics are Prometheus

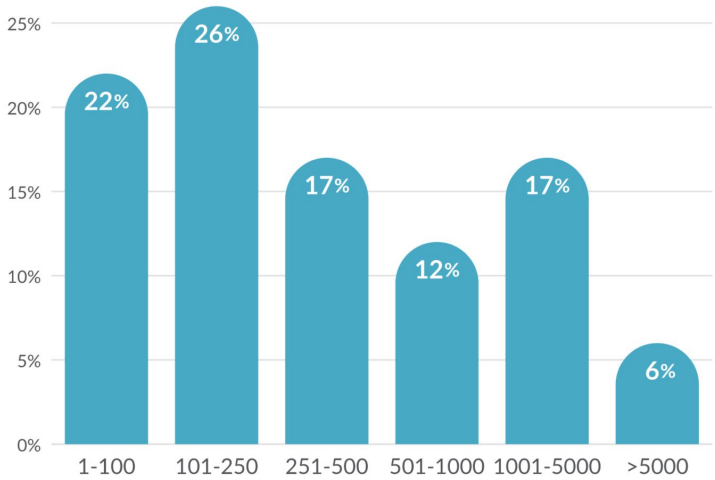
Ref: https://sysdig.com/blog/2022-cloud-native-security-usage-report/

Container World Report





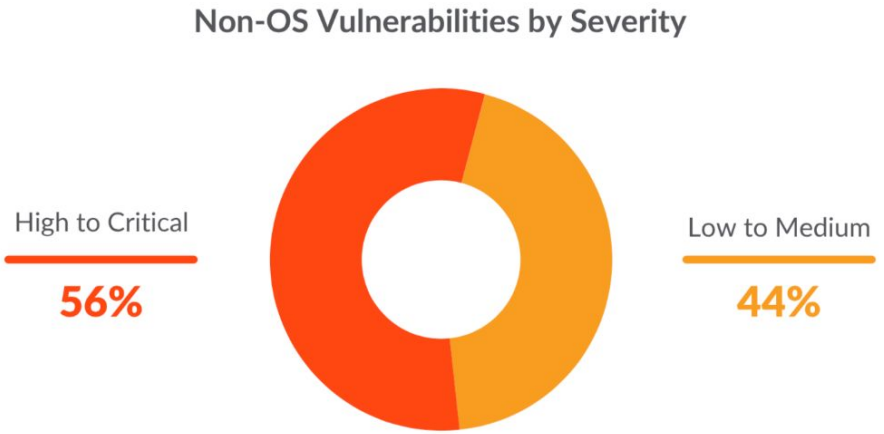


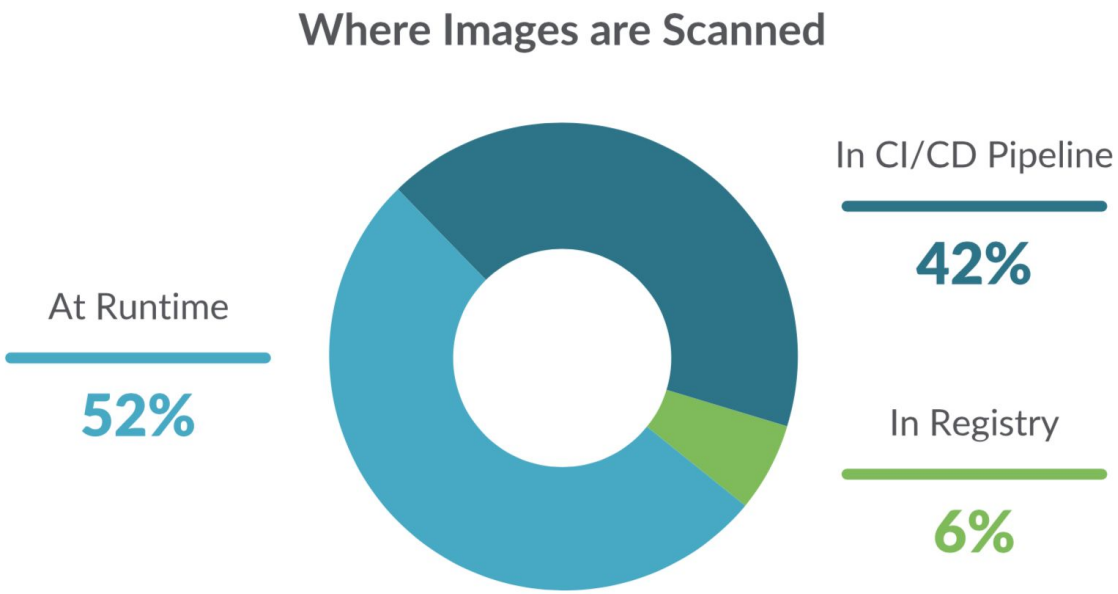


Kubernetes: Production Workload Orchestration

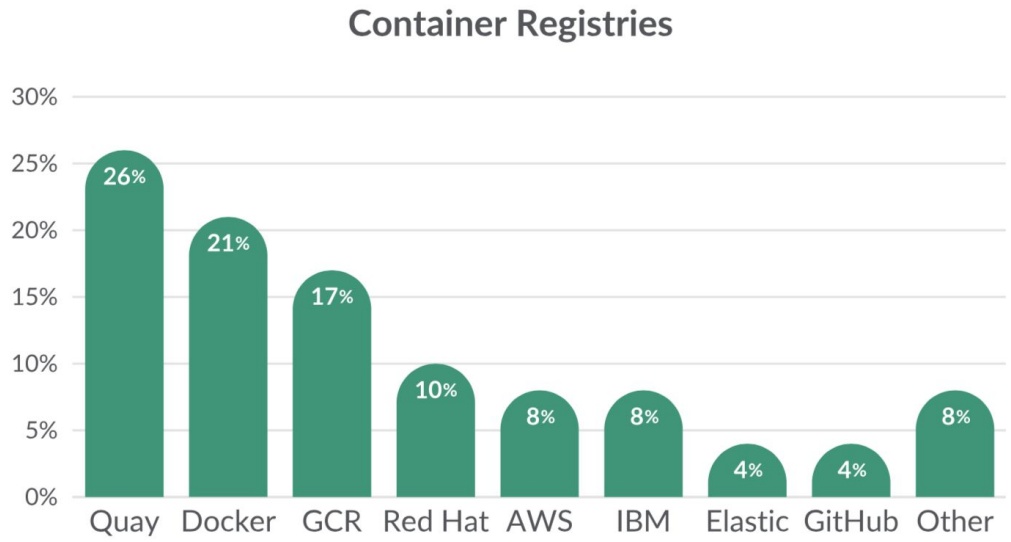
kubernetes by Google

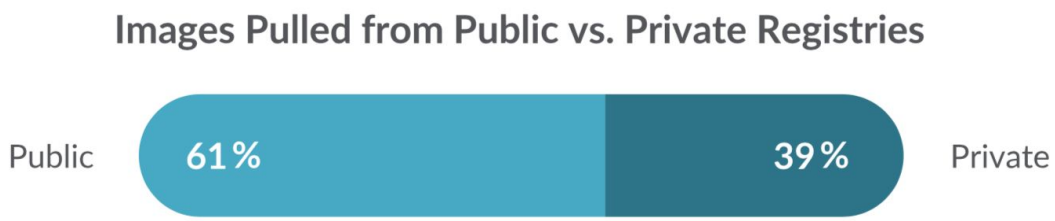
Container World Report





Container World Report

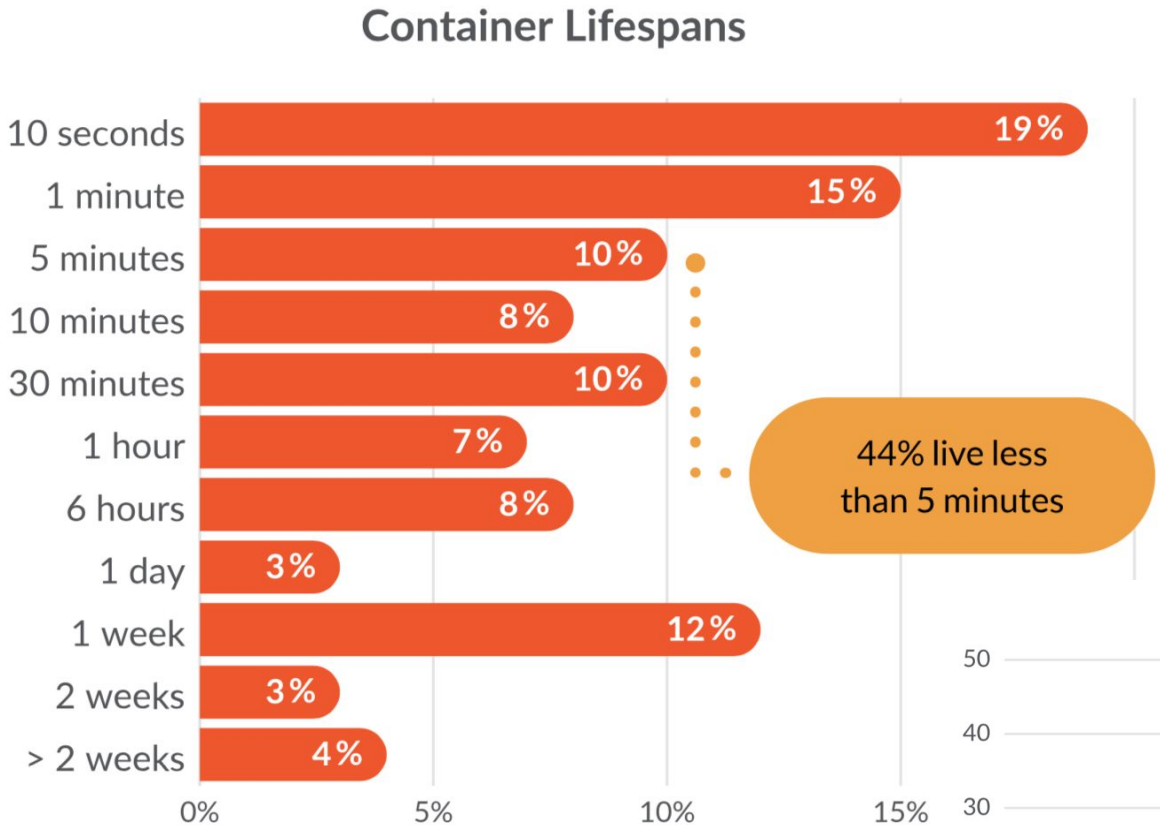


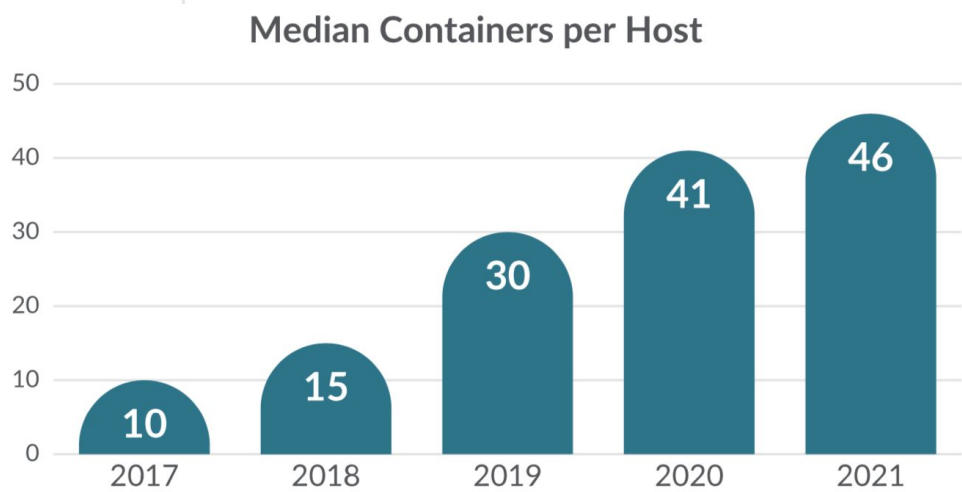


Container World Report

NGINX Part of F5	GO	JMX	redis
65%	57%	37%	30%
PostgreSQL	node JS	mongoDB	elastic
20%	16%	14%	12%
RabbitMQ	APACHE	kafka	CouchDB
12%	11%	8%	6%

Container World Report

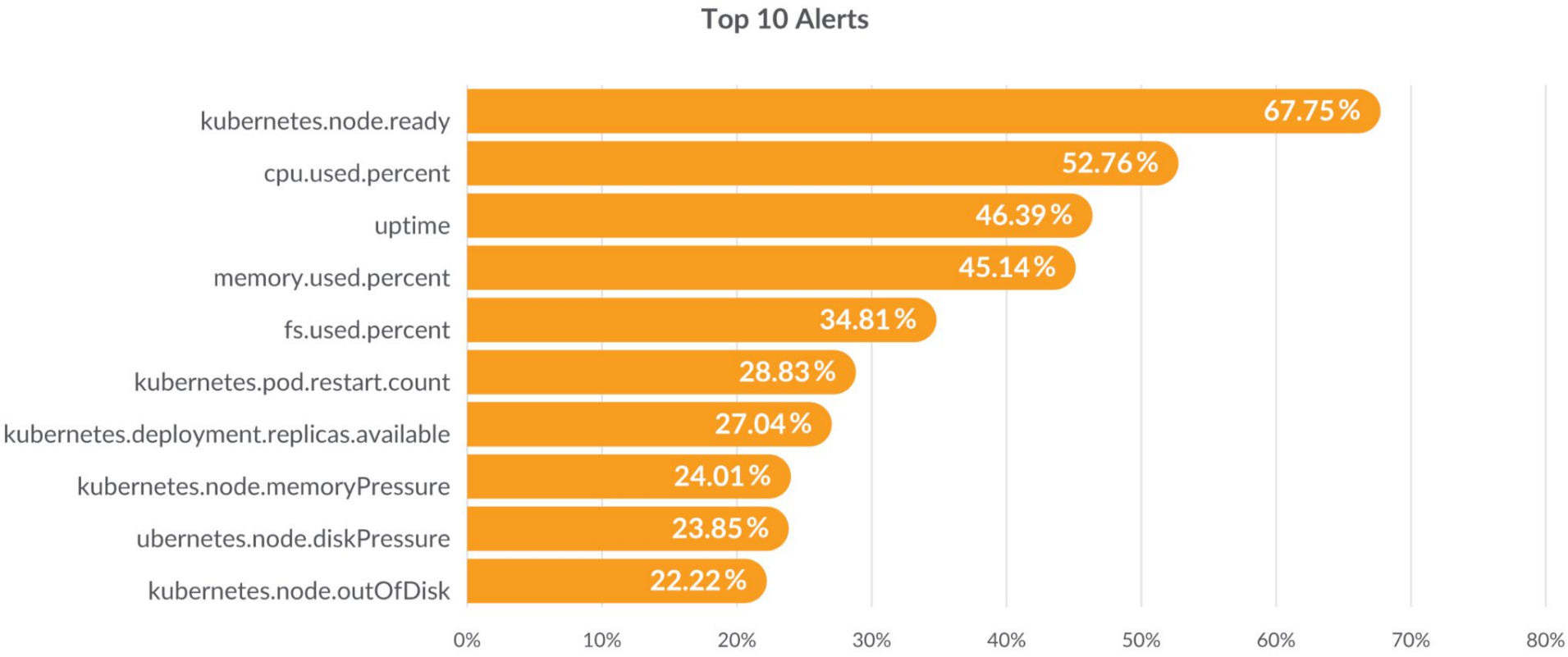




Kubernetes: Production Workload Orchestration

kubernetes by Google

Container World Report



Workshop: Access Container on AWS

+

amazon web services™

Docker on AWS

kubernetes by Google

Workshop: Access Container on AWS

https://labs.play-with-docker.com

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

Contribute

Play

A simple, interac

03:59:39

CLOSE SESSION

Instances

+ ADD NEW INSTANCE

192.168.0.13
node1

c8ecrt5m_c8ecrudmrepg00av6lig

IP
192.168.0.13

OPEN PORT

Memory
0.80% (32.12MiB / 3.906GiB)

CPU
0.60%

SSH
ssh ip172-18-0-54-c8ecrt5mrepg00av6li0@direct.labs.pla

DELETE

EDITOR

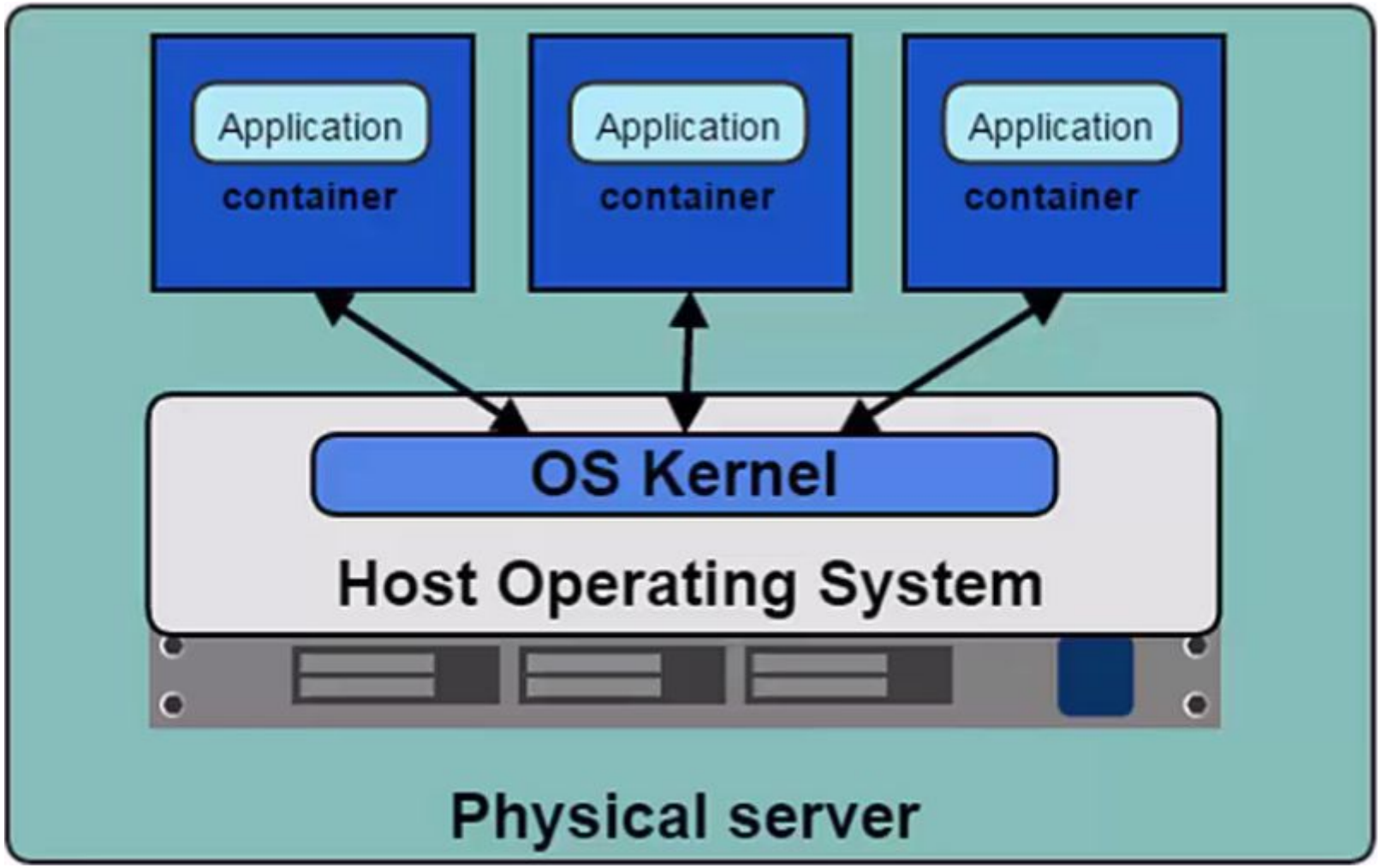
```
# completely the user's responsibilites.                    #
#                                                           #
# The PWD team.                                             #
#############################################################
[node1] (local) root@192.168.0.13 ~
$ $ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[node1] (local) root@192.168.0.13 ~
$ docker version
Client: Docker Engine - Community
 Version:           20.10.0
 API version:       1.41
 Go version:        go1.13.15
 Git commit:        7287ab3
 Built:             Tue Dec  8 18:54:00 2020
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true
```

Kubernetes: Production Workload Orchestration

kubernetes by Google

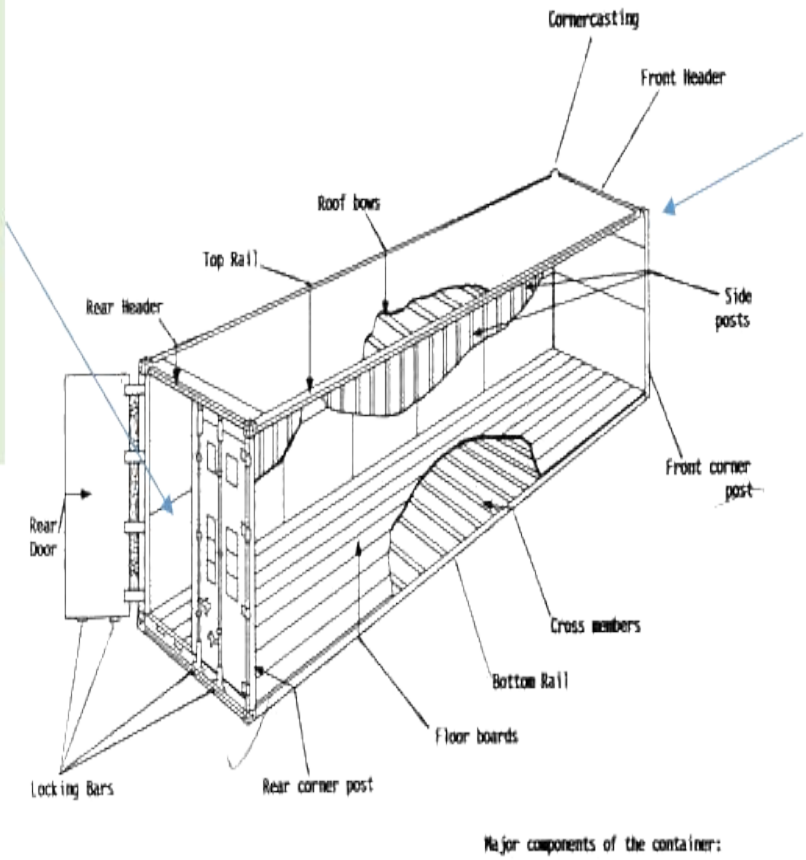
Container Principle

Container Architecture



Separate of Concern

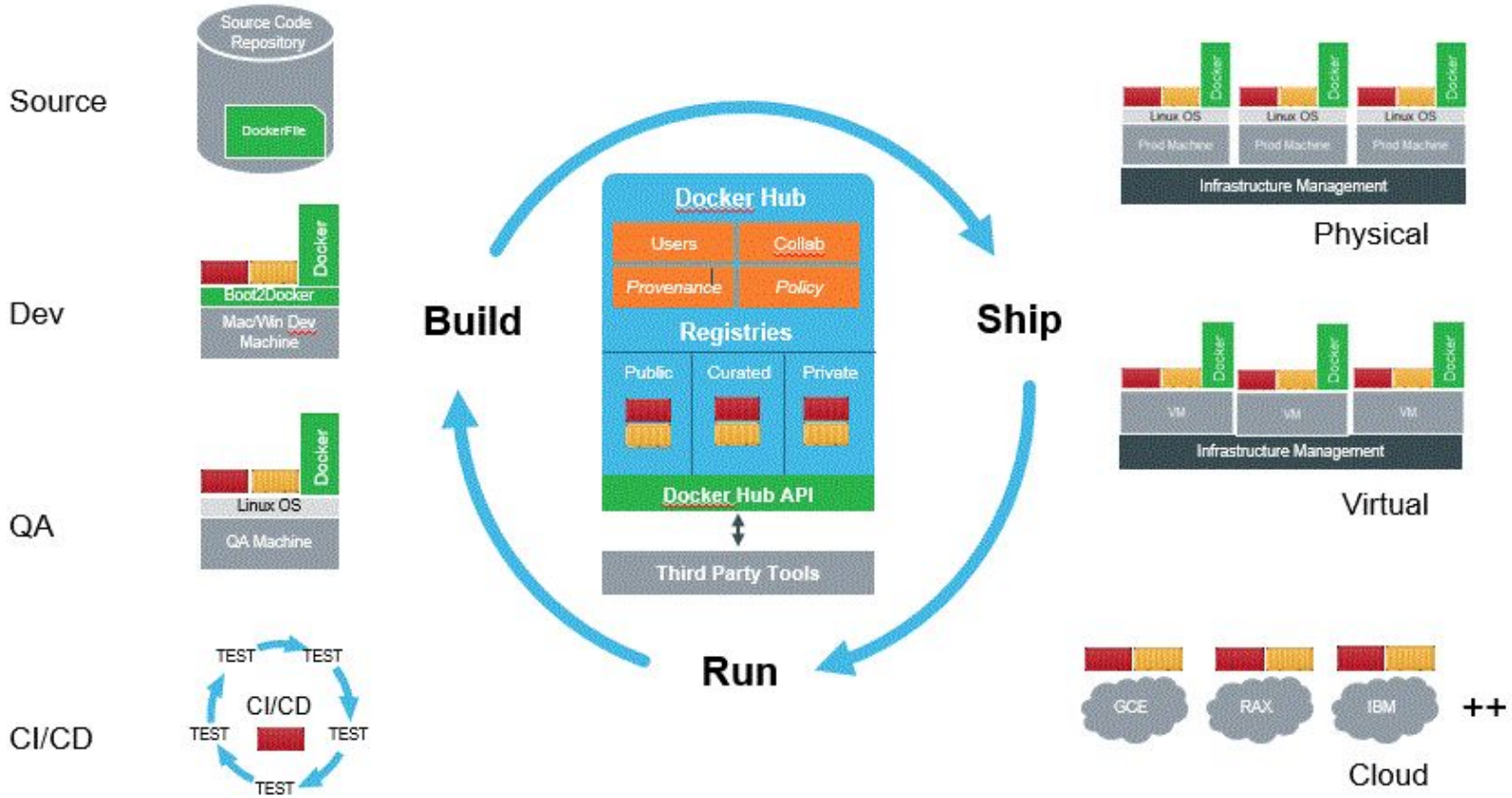
- Dan the Developer
  - Worries about what's "inside" the container
    - His code
    - His Libraries
    - His Package Manager
    - His Apps
    - His Data
  - All Linux servers look the same



- Oscar the Ops Guy
  - Worries about what's "outside" the container
    - Logging
    - Remote access
    - Monitoring
    - Network config
  - All containers start, stop, copy, attach, migrate, etc. the same way

Container Build-Ship-Run

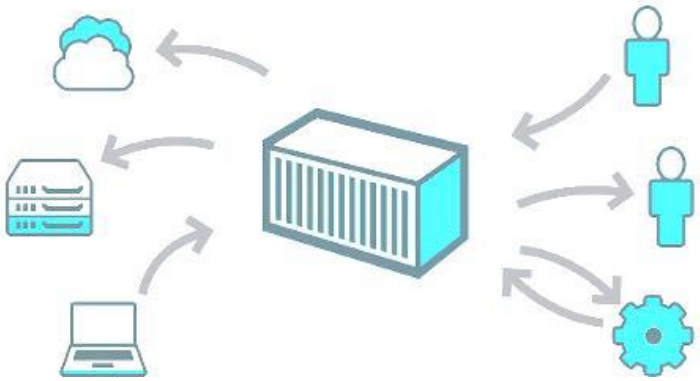
- Build-Ship-Run



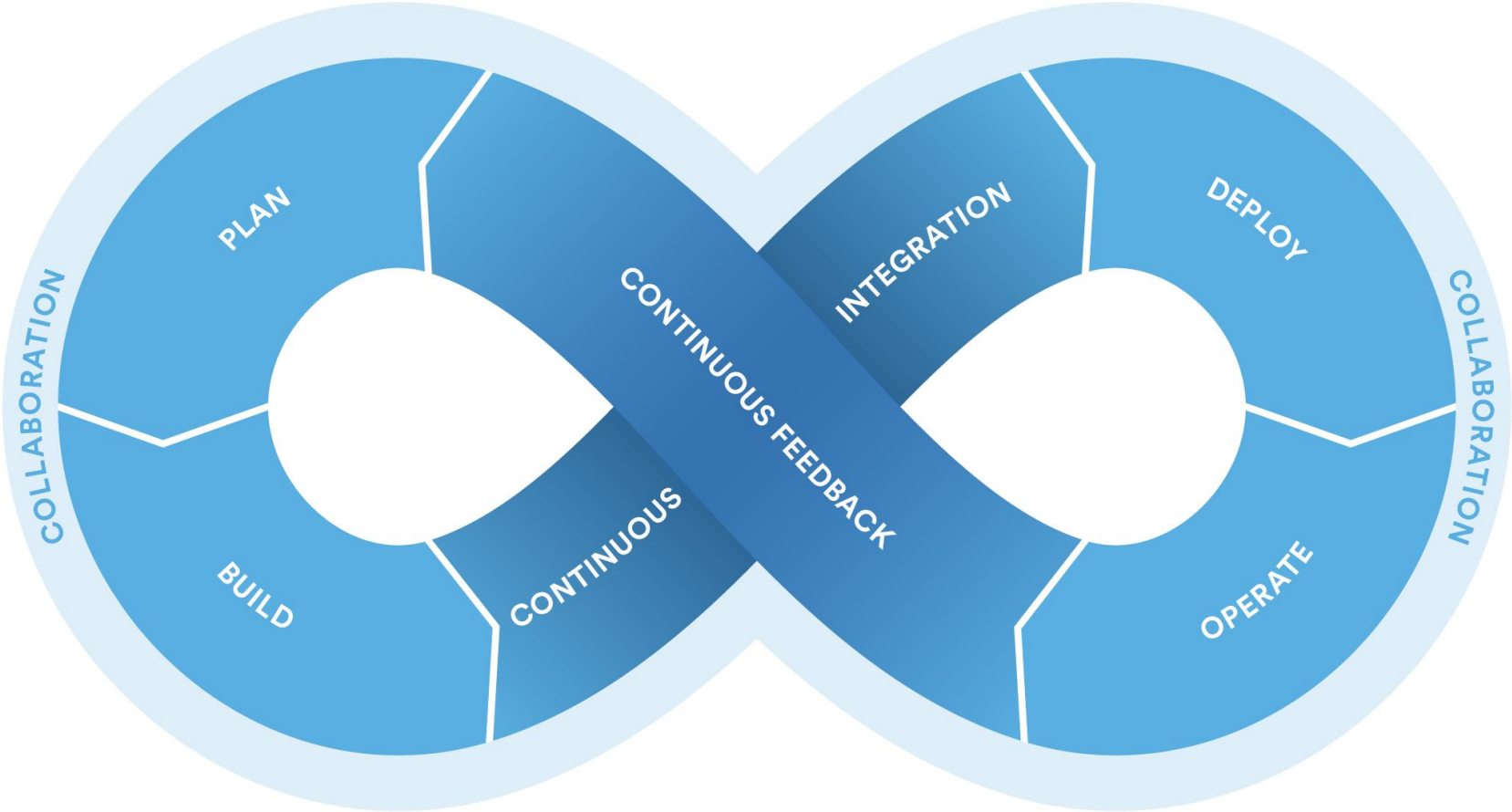
Container Life Cycle

- Developer
  - Develop program and build package (build)
  - Test run application (run)
  - Package final image for deployment (commit)

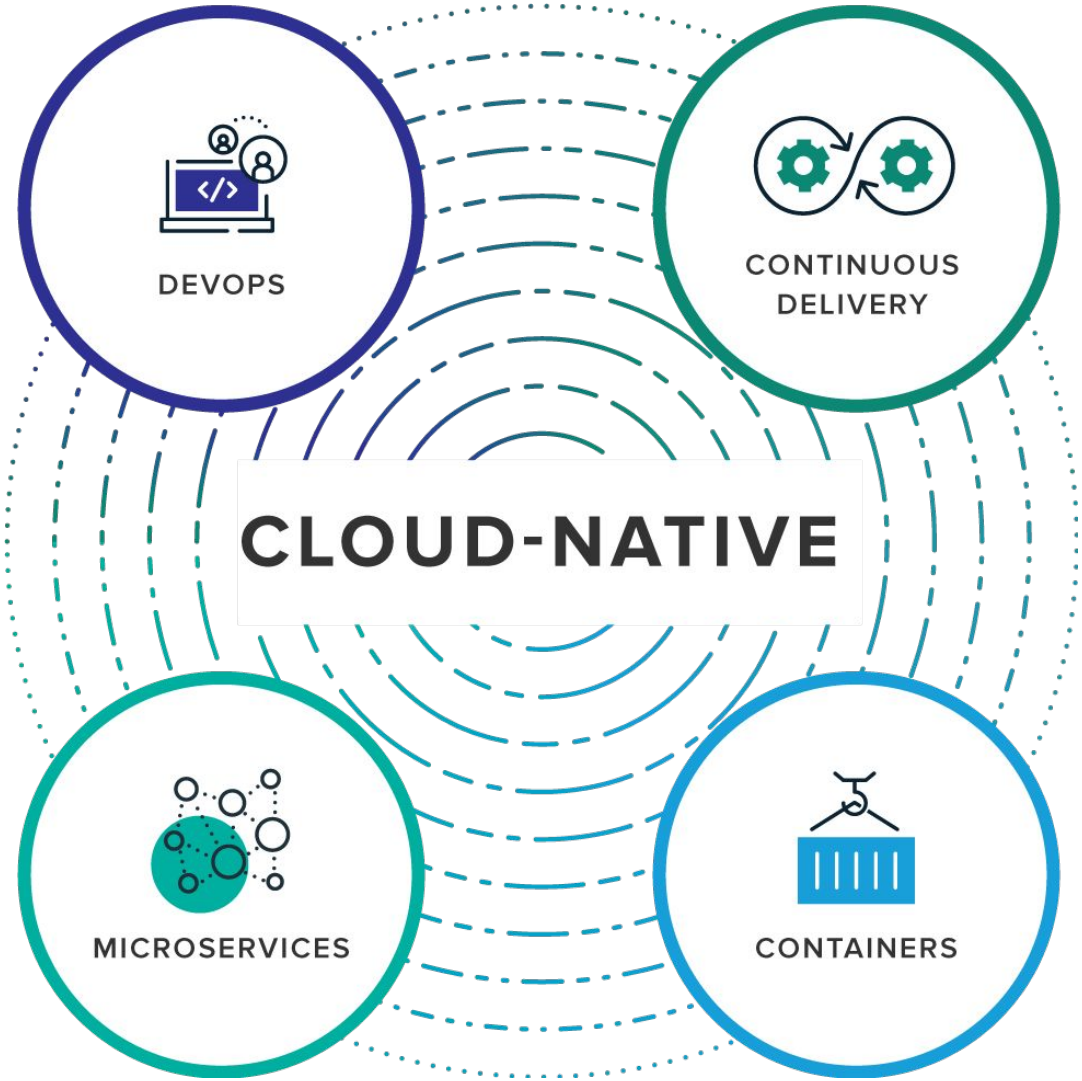
- Operation
  - Pull image and Run on production (run)
  - Stop existing container and start new container version (stop, kill)
  - Remove unused container from production (rm, rmi)



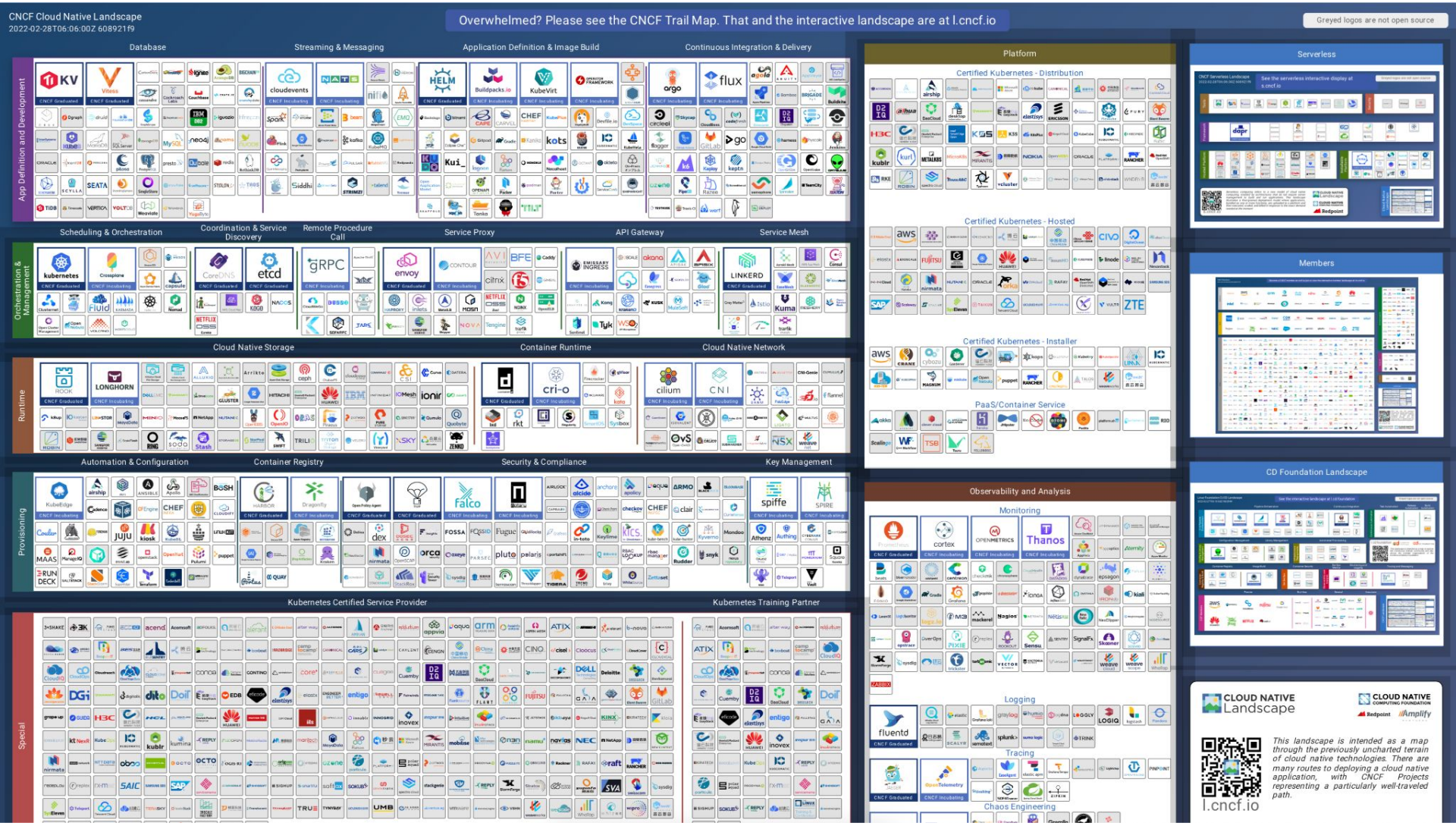
Container Life Cycle



Cloud Native Architecture



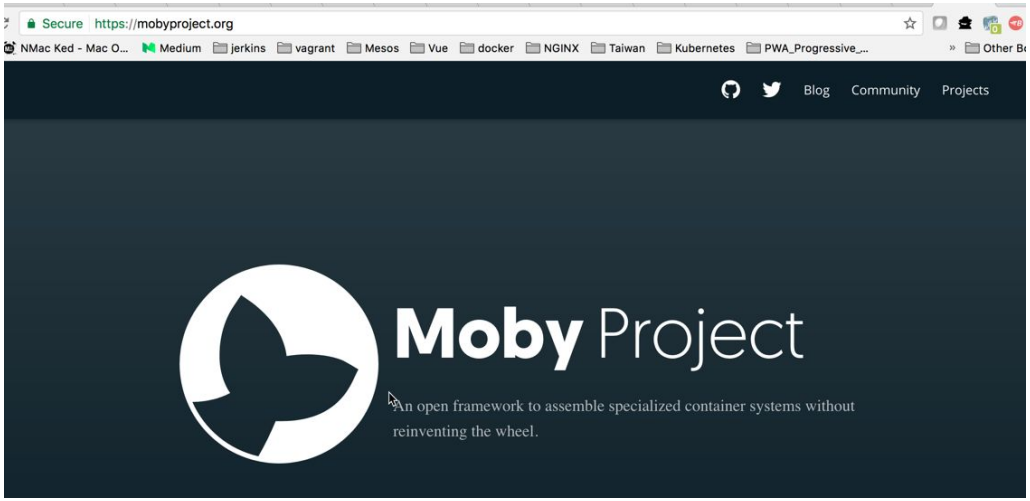
Cloud Native Computing Foundation

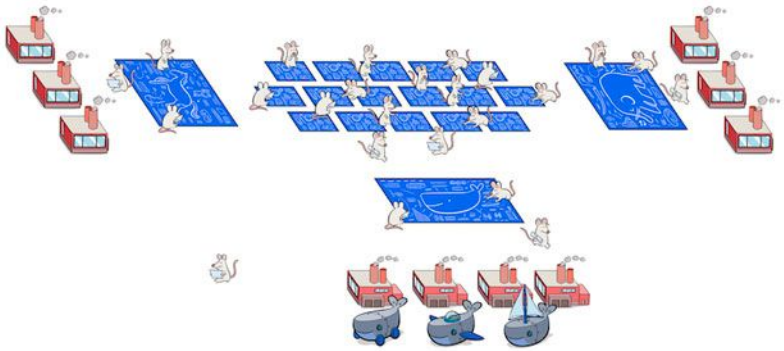


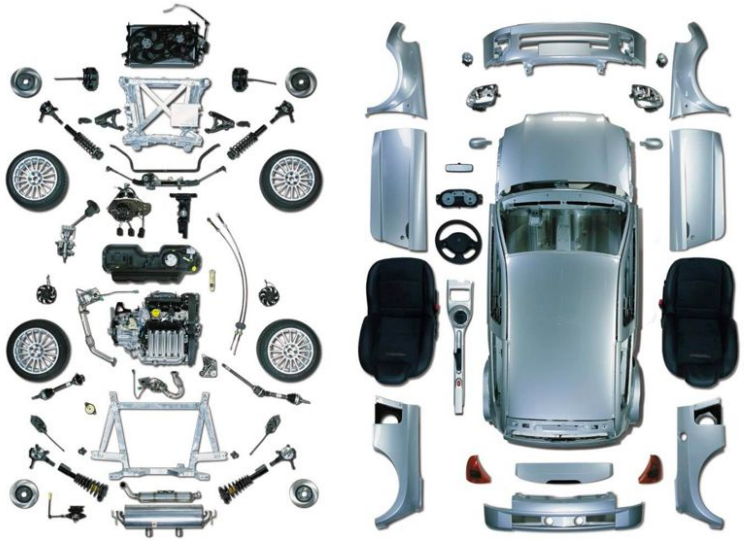
Kubernetes: Production Workload Orchestration

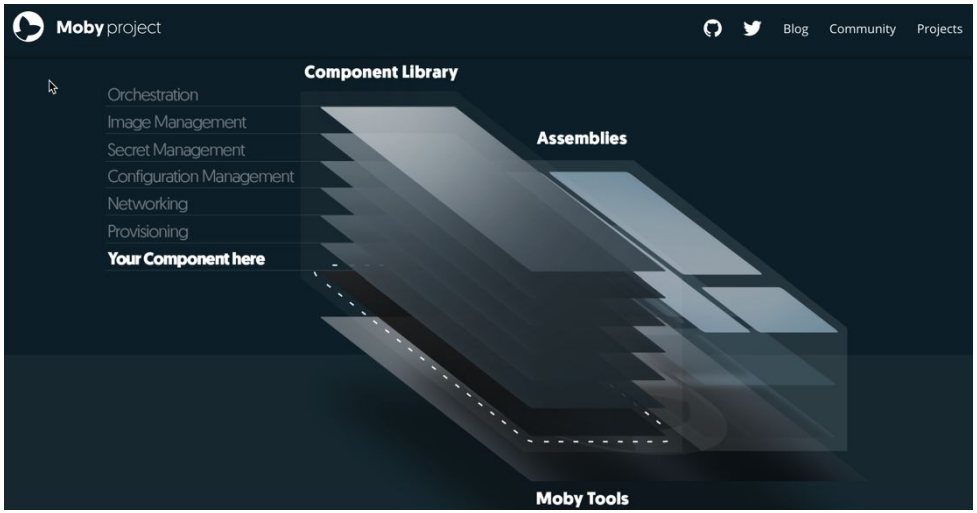
kubernetes by Google

Moby Project for Kubernetes









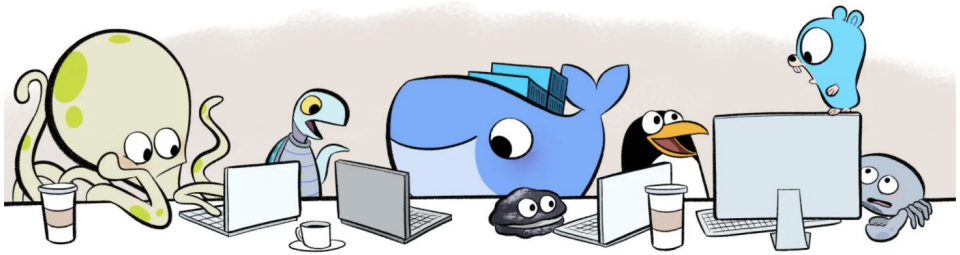
Kubernetes: Production Workload Orchestration

kubernetes by Google

Subscription warning

Docker is Updating and Extending Our Product Subscriptions

SCOTT JOHNSTON
Aug 31 2021



Docker is used by millions of developers to build, share, and run any app, anywhere, and 55% of professional developers use Docker every day at work. In these work environments, the increase in outside attacks on software supply chains is accelerating developer demand for Docker's trusted content, including Docker Official Images and Docker Verified Publisher images. Finally, the rapid global growth in developers – to an estimated 45 million by 2030 – pushes us to scale sustainably so we may continue to provide an innovative, free Docker experience that developers love.

To meet these challenges, today we're announcing updates and extensions to our product subscriptions: **Personal, Pro, Team, and Business.** These updated product subscriptions provide the productivity and collaboration developers rely on with the scale, security, and trusted content businesses require, and do so in a manner sustainable for Docker.

What you need to know:

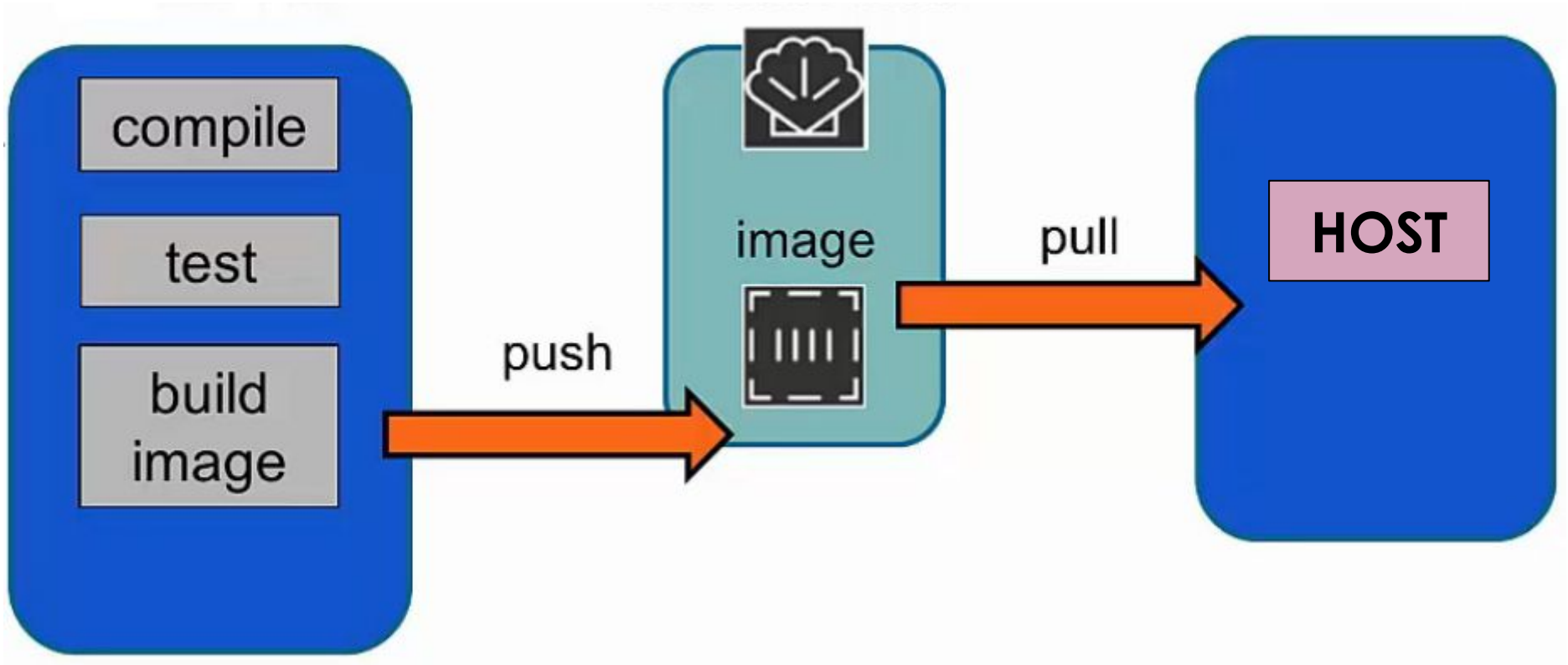
- We're introducing a **new** product subscription, Docker Business, for organizations using Docker at scale for application development and require features like secure software supply chain management, single sign-on (SSO), container registry access controls, and more.
- Our Docker Subscription Service Agreement includes a change to the terms for **Docker Desktop**:
  - Docker Desktop **remains free** for small businesses (fewer than 250 employees AND less than $10 million in annual revenue), personal use, education, and non-commercial open source projects.
  - It requires a paid subscription (**Pro**, **Team** or **Business**), starting at $5 per user per month, for professional use in larger businesses. You may directly purchase here, or share this post and our solution brief with your manager.
  - While the effective date of these terms is August 31, 2021, there is a grace period until January 31, 2022 for those that require a paid subscription to use Docker Desktop.
- Docker Pro, Docker Team, and Docker Business subscriptions **include** commercial use of Docker Desktop.
- The **existing** Docker Free subscription has been renamed Docker Personal. **No changes** to Docker Engine or any upstream open source Docker or Moby project.
- *Check out our FAQ or more information.*

Personal	Pro	Team	Business
1	1	5+	50+
Ideal for individual developers, education, open source communities, and small businesses.	Ideal for individual developers	Ideal for development teams	Ideal for medium and large businesses
$0	$5 /month	$7 /user/month minimum 5 seats	$21 /user/month
Includes Docker Desktop	Includes Docker Desktop	Includes Docker Desktop	Includes Docker Desktop

kubernetes by Google

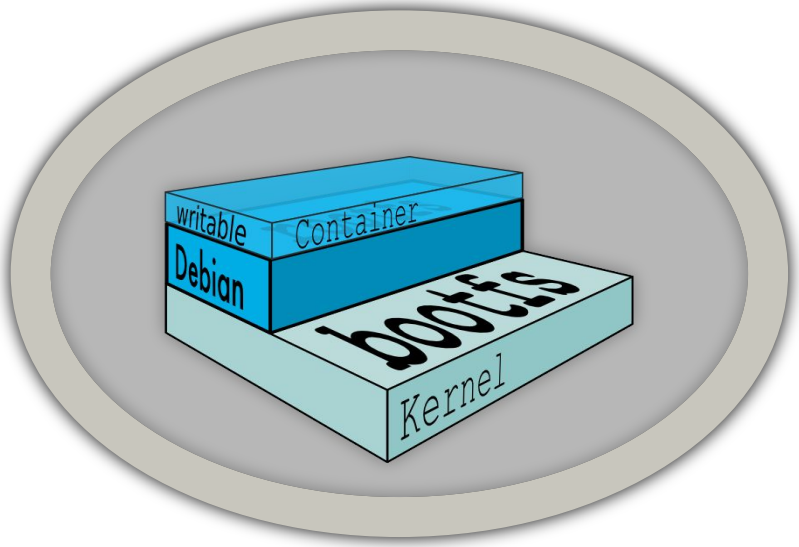
Image, Repository & Tag, container

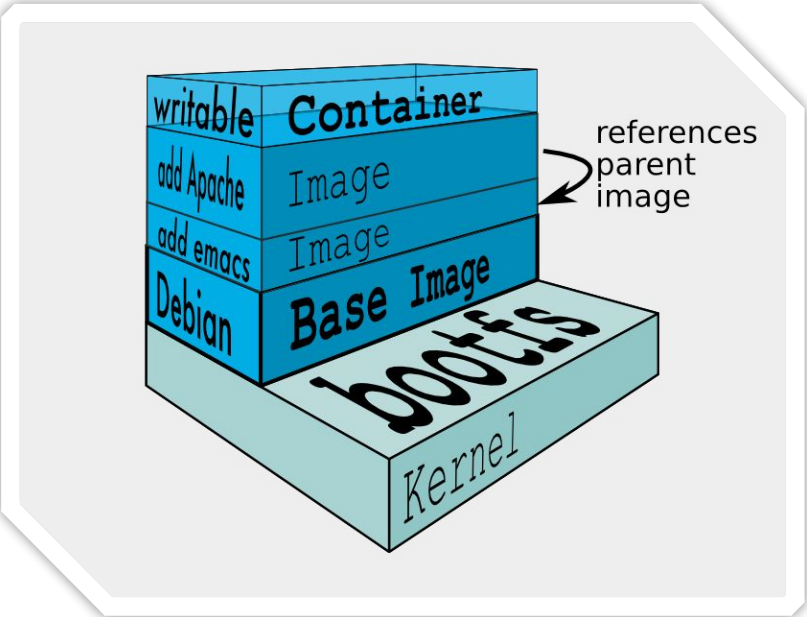
Docker Deployment



Container Image

- Image is “template” that was created by other or us for run the container
- Image is “read only”
- Store image on repository (Cloud / On-Prem)





Workshop: Pull Image

Pull image from hub.docker.com to lab machine

docker image pull labdocker/alpine:3.13.6

docker image pull labdocker/alpineweb:latest

docker image pull labdocker/cadvisor:0.40.0

Compare with Ubuntu image

docker images/docker image ls

```
ubuntu@ip-10-0-1-90:~$ docker image ls
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
labdocker/alpineweb   latest    75f4bf61fa65   51 minutes ago   62.1MB
labdocker/alpine      3.13.6    12adea71a33b   2 days ago       5.61MB
labdocker/ubuntu      latest    fb52e22af1b0   3 days ago       72.8MB
labdocker/cadvisor    0.40.0    1ed215a61956   8 weeks ago      86.9MB
ubuntu@ip-10-0-1-90:~$
```

Workshop: Pull Image

Check image detail by command: “history”

docker image history labdocker/alpine:3.13.6

docker image history labdocker/alpineweb:latest

```
ubuntu@ip-10-0-1-90:~$ docker image history labdocker/alpine:3.13.6
IMAGE          CREATED      CREATED BY                                      SIZE      COMMENT
12adea71a33b   2 days ago   /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      2 days ago   /bin/sh -c #(nop) ADD file:ecdfb91a737d6c292…   5.61MB
ubuntu@ip-10-0-1-90:~$     docker image history labdocker/alpineweb:latest
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
75f4bf61fa65   53 minutes ago   /bin/sh -c #(nop)  EXPOSE 3000                  0B
<missing>      53 minutes ago   /bin/sh -c #(nop)  HEALTHCHECK &{["CMD-SHELL…   0B
<missing>      53 minutes ago   /bin/sh -c #(nop)  ENTRYPOINT ["node" "hello…   0B
<missing>      53 minutes ago   /bin/sh -c #(nop) WORKDIR /nodejs               0B
<missing>      53 minutes ago   /bin/sh -c #(nop) COPY file:7a1f27eb22f7a835…   263B
<missing>      53 minutes ago   /bin/sh -c mkdir /nodejs                        0B
<missing>      53 minutes ago   /bin/sh -c apk update &&     apk add nodejs …   56.5MB
<missing>      2 hours ago      /bin/sh -c #(nop)  ENV NODE_VERSION=v14.15.4…   0B
<missing>      2 hours ago      /bin/sh -c #(nop)  LABEL Description=NodeJS/…   0B
<missing>      2 hours ago      /bin/sh -c #(nop)  LABEL maintainer=Praparn …   0B
<missing>      2 days ago       /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      2 days ago       /bin/sh -c #(nop) ADD file:ecdfb91a737d6c292…   5.61MB
ubuntu@ip-10-0-1-90:~$
```

Container Image

- Check image detail on container server

docker images /docker image ls

```
ubuntu@ip-10-0-1-90:~$ docker image ls
REPOSITORY                TAG              IMAGE ID       CREATED          SIZE
labdocker/linenotify      bot_v2           7f073d0abe25   8 minutes ago    86.4MB
labdocker/alpineweb       latest           75f4bf61fa65   54 minutes ago   62.1MB
labdocker/alpine          3.13.6           12adea71a33b   2 days ago       5.61MB
labdocker/mariadb         10.5.12          b7b798989225   3 days ago       407MB
labdocker/ubuntu          latest           fb52e22af1b0   3 days ago       72.8MB
labdocker/cadvisor        0.40.0           1ed215a61956   8 weeks ago      86.9MB
labdocker/cluster         webservicelite   837c8f41c918   4 years ago      82.6MB
labdocker/alpinepython    2.7-onbuild      0f4df961b16b   4 years ago      475MB
ubuntu@ip-10-0-1-90:~$
```

- Tag image from existing to new one

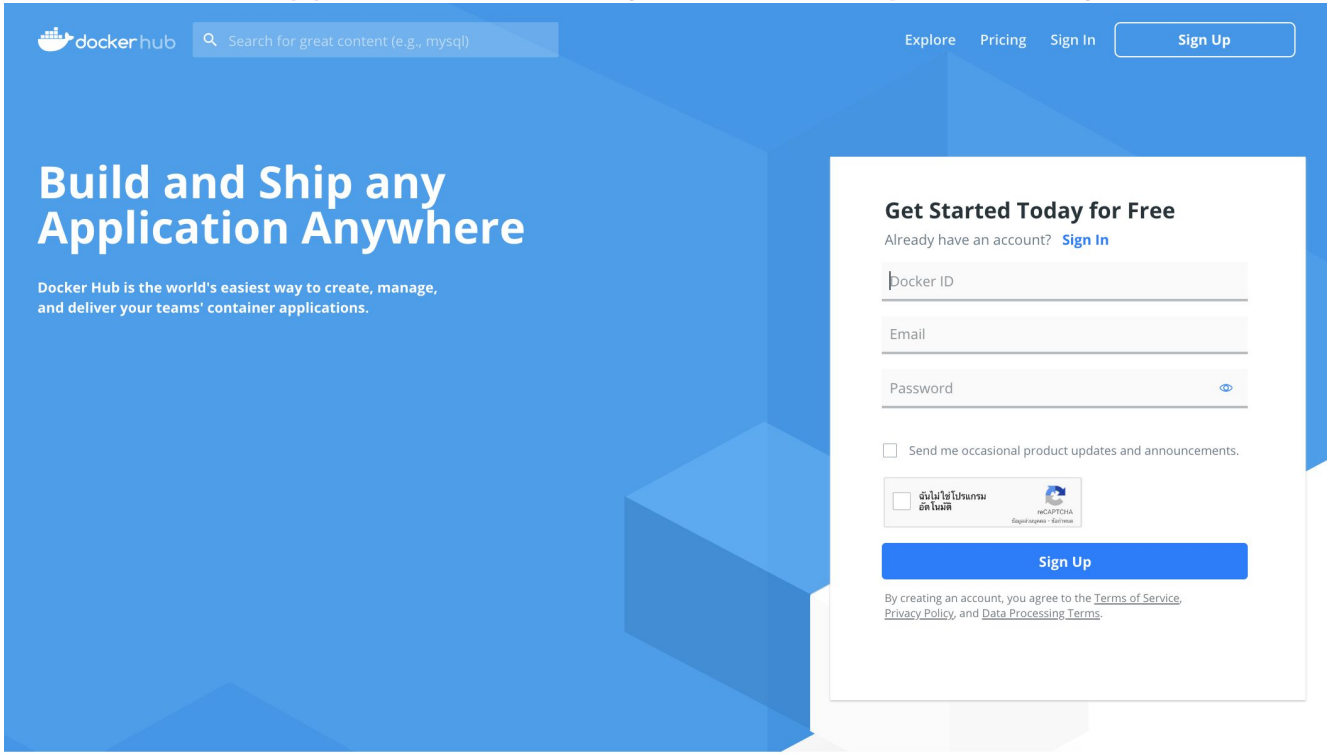
docker image tag <image id> <acc name/imagename: version>

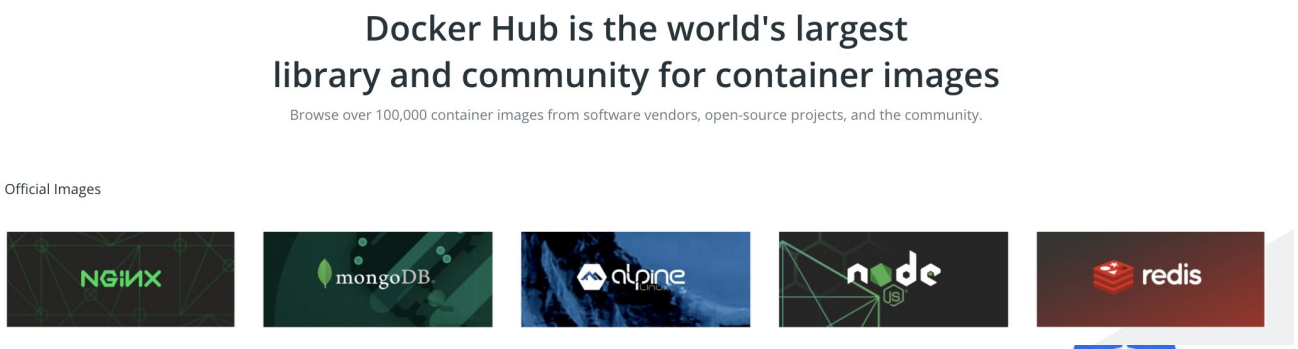
Ex: docker image tag 14f89d0e6257 labdocker/alpinelab:1.0

Repository (Registry)

- Component for store the image as centralize and distribution to all container server
- Support image versioning system
- Give the instruction for open-source user to pull and run container
- Docker already have service repository on cloud via https://hub.docker.com
- Official image from product owner
- Free registry without cost (Limit push/pull request/6 hours)
- Private repository (registry, trust registry, harbor etc)
- Other repository also can use (Quay.io, ECR etc)

Repository (Registry)





Kubernetes: Production Workload Orchestration

kubernetes by Google

Repository (Registry)

docker

Why Docker? Products Developers Pricing Company Sign In Get Started

Understanding Docker Hub Rate Limiting

Learn More

On November 20, 2020, rate limits anonymous and free authenticated use of Docker Hub went into effect. Anonymous and Free Docker Hub users are limited to 100 and 200 container image pull requests per six hours.

If you are affected by these changes you will receive this error message:

```
ERROR: toomanyrequests: Too Many Requests.
```

OR

```
You have reached your pull rate limit. You may increase the limit by authenticating and upgrading: https://www.docker.com/increase-rate-limits.
```

To increase your pull rate limits you can upgrade your account to a Docker Pro or Team subscription.

The rate limits of 100 container image requests per six hours for anonymous usage, and 200 container image requests per six hours for free Docker accounts are now in effect. Image requests exceeding these limits will be denied until the six hour window elapses.

NOTE: Docker Pro and Docker Team accounts continue to have unlimited access to pull container images from Docker Hub.

Related Resources

- Rate Limiting Documentation
- Resource Consumption Update FAQ
- Blog post on Resource Consumption and Image Retention Policy Updates
- Docker Subscription Pricing Page
- Pricing FAQ
- Guidance for Setting Up a Mirror
- Understanding Inner Loop Development and Pull Rates
- Docker Partner Information Request for Rate Limiting Exclusions

Repository (Registry)

google/cadvisor

Pulls 10M+

By google • Updated 12 hours ago

Analyzes resource usage and performance characteristics of running containers.

Container

Overview Tags

cAdvisor

cAdvisor (Container Advisor) provides container users an understanding of the resource usage and performance characteristics of their running containers. It is a running daemon that collects, aggregates, processes, and exports information about running containers. Specifically, for each container it keeps resource isolation parameters, historical resource usage, and histograms of complete historical resource usage. This data is exported by container and machine-wide.

https://github.com/google/cadvisor

Official cAdvisor releases are built on Linux and exported over a scratch image, this guarantees a small image size.

Dockefile: https://github.com/google/cadvisor/blob/master/deploy/Dockerfile

Each cAdvisor version is tagged. We also have 2 tags for the beta and stable track:

- **latest:** Latest stable build, this is the latest officially supported releases.
- **canary:** Images built from HEAD periodically. Potentially unstable!

We also have an Automated Build canary release of cAdvisor which is continuously built from HEAD. This can be found in the google/cadvisor-canary image. It is not recommended for production use due to its size and volatility.

Docker Pull Command

```
docker pull google/cadvisor
```

Owner

google

Repository (Registry)

- download image from registry server

```
docker image pull <account name/image name: tags>
```

```
Ex: docker image pull labdocker/alpine:latest
```

- Upload image to registry server
  - Login

```
docker login <url> -u <username> -p <password> -e <email>
```

```
Ex: docker login -u labdocker –p xxxxxxx –e xxxx@xxx.xxx
```

```
Ex: docker login 10.38.7.248:8080 –u labdocker
```

Repository (Registry)

- Push image to registry server

```
docker image push<account name/image name: tags>
```

Ex: docker image push labdocker/alpinelab:1.0

- Logoff to registry server

```
docker logoff
```

**Docker 1.10 (and upper) will push image as parallel (each layer will upload as independency)*

Repository (Registry)

- (Offline) Export image as offline

```
docker image save -o xxx.tar image name
```

Ex: docker image save -o test.tar labdocker/alpinelab:1.0

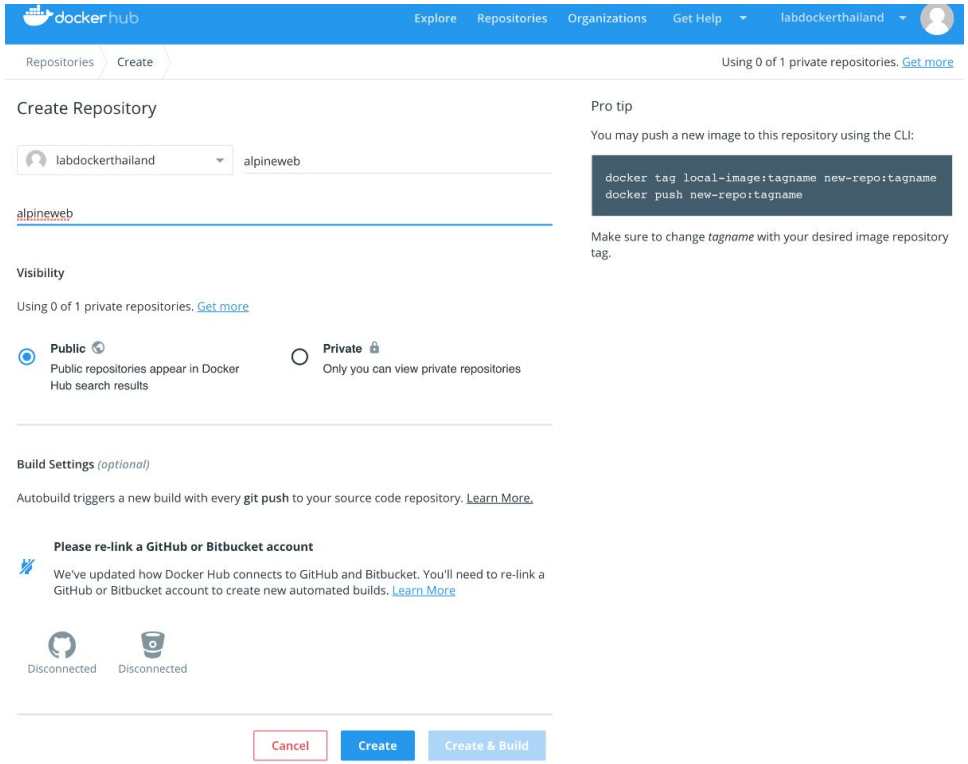
- (Offline) Import image as offline

```
docker image load -i xxx.tar
```

Ex: docker image load -i test.tar

Workshop: Create & Push Image

- On the workshop we will register account to http://hub.docker.com and tag image for push image
- Create repository "Alpine Web"



Workshop: Build & Push Image

- Check image on container server

```
docker image ls
```

- tag image as <accountname>/alpineweb:latest

```
docker image tag labdocker/alpineweb:latest \
<accountname>/alpineweb:latest
```

- Login to hub.docker.com withusername/password

```
docker login –u <xxxx>
```

- Push image alpine as tag upload to repository server

```
docker image push <accountname>/alpineweb:latest
```

Workshop: Build & Push Image

- Test to export image as .tar file

```
docker image save -o ~/alpineweb.tar
<account>/alpineweb:latest
```

- Test to delete image and re import as tar file

```
docker image rm <account>/alpineweb:latest
docker image load -i ~/alpineweb.tar
```

Other service in hub.docker.com

Access the world's largest library of container images

Official Images

NGINX, mongoDB, alpine, node, redis

couchbase Official 10M+
ubuntu Official 10M+
busybox Official 10M+
mysql Official 10M+
postgres Official 10M+
hello-world Official 10M+
registry Official 10M+
traefik Official 10M+
docker Official 10M+
mariadb Official 10M+

docker hub | Search for great content (e.g., mysql) | Explore | Repositories | Organizations | Get Help | labdockerthailand

ubuntu
Docker Official Images
Ubuntu is a Debian-based Linux operating system based on free software.
10M+
Container | Linux | 386 | ARM 64 | ARM | x86-64 | IBM Z | PowerPC 64 LE | Base Images | Operating Systems | Official Image

Linux - ARM (latest)
Copy and paste to pull this image
```
docker pull ubuntu
```
View Available Tags

DESCRIPTION | REVIEWS | TAGS

Supported tags and respective `Dockerfile` links

- `18.04`, `bionic-20190204`, `bionic`, `latest` (*bionic/Dockerfile*)
- `18.10`, `cosmic-20190131`, `cosmic`, `rolling` (*cosmic/Dockerfile*)
- `19.04`, `disco-20190204`, `disco`, `devel` (*disco/Dockerfile*)
- `14.04`, `trusty-20190122`, `trusty` (*trusty/Dockerfile*)
- `16.04`, `xenial-20190122`, `xenial` (*xenial/Dockerfile*)

Be the first to give insight into your experience by rating and reviewing the product.

Add Product Review
Select a product tier
Start Your Review

docker hub | Search for great content (e.g., mysql) | Explore | Repositories | Organizations | Get Help | labdockerthailand

Docker EE | Docker CE | Containers | Plugins

Filters
Docker Certified
- Docker Certified

Images
- Verified Publisher — *Docker Certified And Verified Publisher Content*
- Official Images — *Official Images Published By Docker*

Categories
- Analytics
- Application Frameworks
- Application Infrastructure
- Application Services
- Base Images
- Databases
- DevOps Tools
- Featured Images
- Messaging Services
- Monitoring
- Operating Systems
- Programming Languages
- Security
- Storage

Operating Systems
- Linux
- Windows

Architectures
- ARM
- ARM 64
- IBM POWER

1 - 25 of 2,039,306 available images.
Most Popular

VERIFIED PUBLISHER
Oracle Database Enterprise Edition — DOCKER CERTIFIED
By Oracle • Updated a year ago
Oracle Database 12c Enterprise Edition
Container | Docker Certified | Linux | x86-64 | Databases

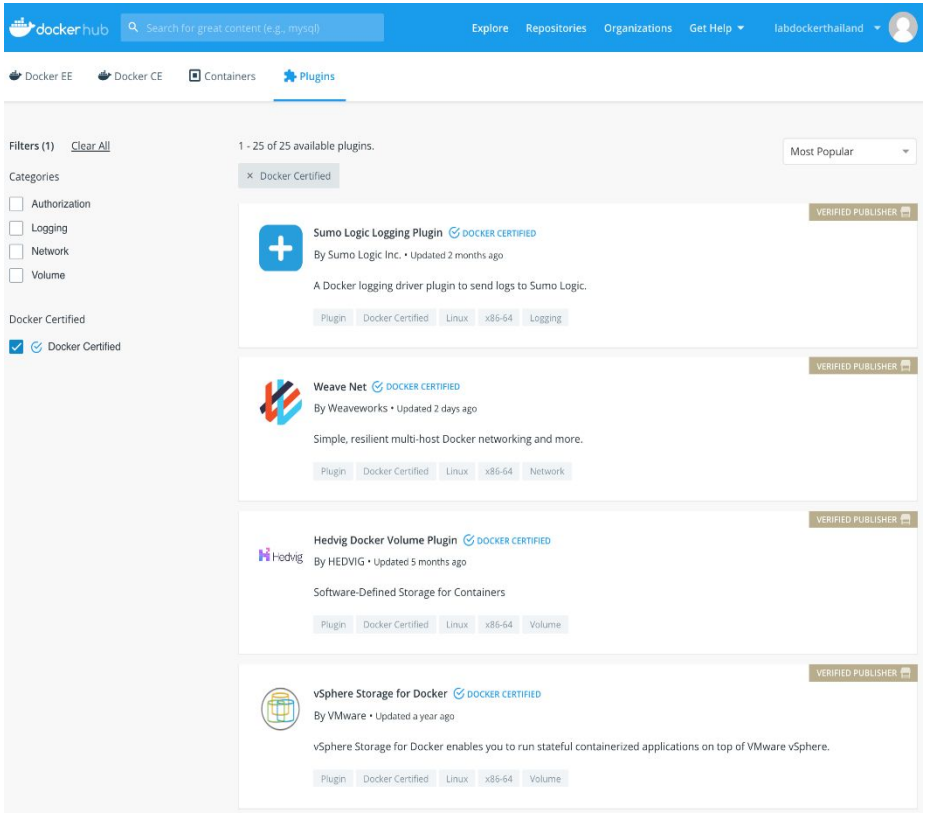
VERIFIED PUBLISHER
Oracle Java 8 SE (Server JRE) — DOCKER CERTIFIED
By Oracle • Updated 2 months ago
Oracle Java 8 SE (Server JRE)
Container | Docker Certified | Linux | x86-64 | Programming Languages

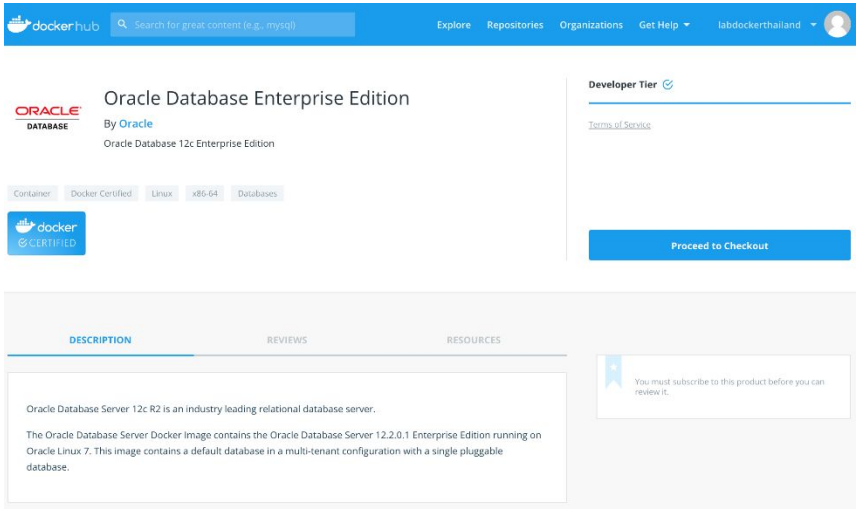
VERIFIED PUBLISHER
MySQL Server Enterprise Edition — DOCKER CERTIFIED
By Oracle • Updated 3 months ago
The world's most popular open source database system
Container | Docker Certified | Linux | x86-64 | Databases

VERIFIED PUBLISHER
Oracle WebLogic Server — DOCKER CERTIFIED
By Oracle • Updated a month ago
Oracle WebLogic Server
Container | Docker Certified | Linux | x86-64 | Application Frameworks | Application Infrastructure

OFFICIAL IMAGE
couchbase
Updated 22 minutes ago
10M+ Downloads 375 Stars
Couchbase Server is a NoSQL document database with a distributed architecture.
Container | Linux | x86-64 | Storage | Application Frameworks

Other service in hub.docker.com





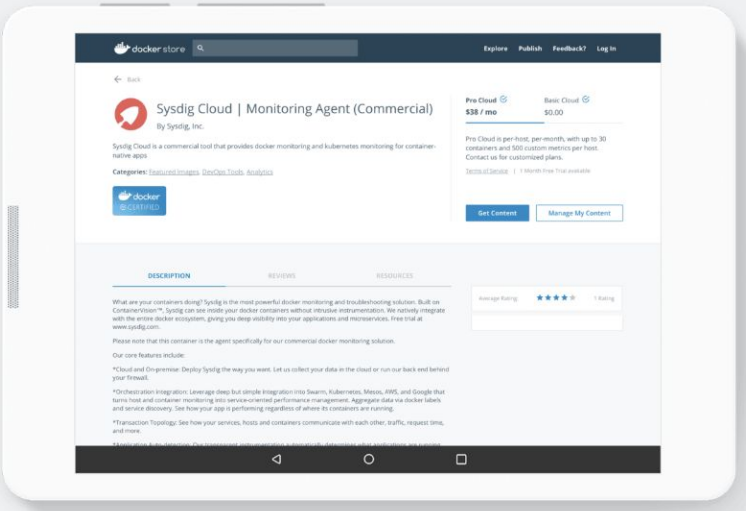
Other service in hub.docker.com

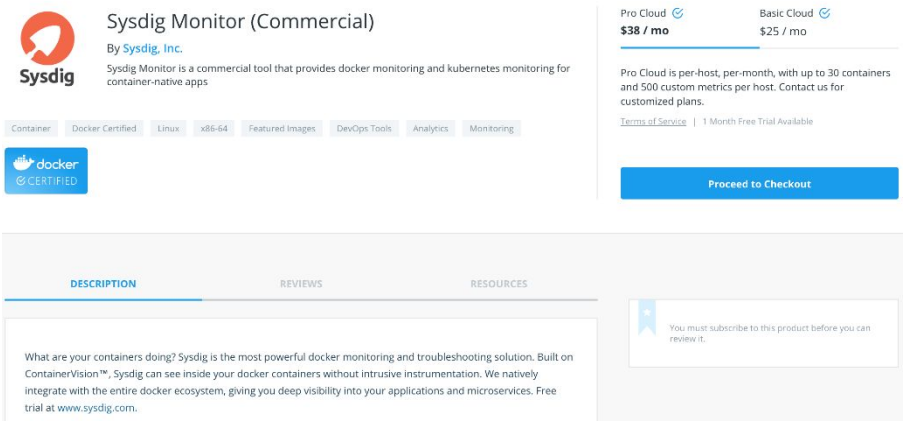
PUBLISHER PROGRAM

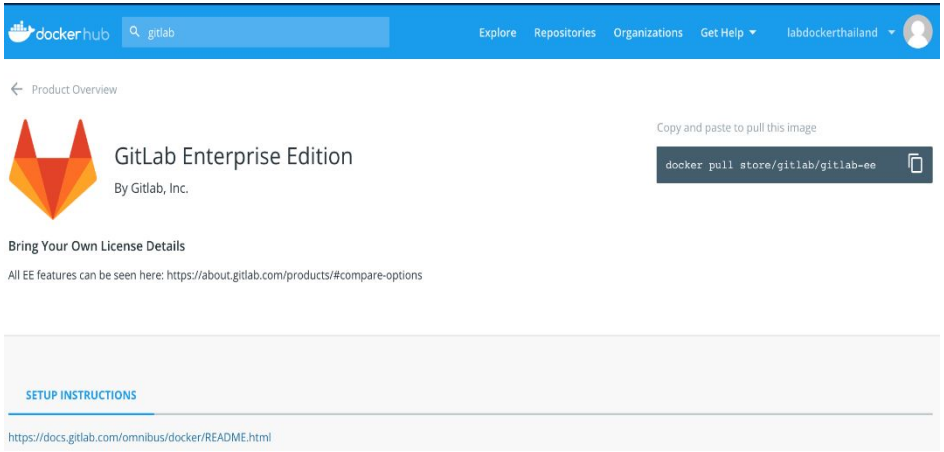
Deliver your business through Docker Hub

Package and publish apps and plugins as containers in Docker Hub for easy download and deployment by millions of Docker users worldwide.

Apply To Publish | Learn More







kubernetes by Google

Status.docker.com

docker system status

Docker system status real-time view

SUBSCRIBE

All Systems Operational Updated a few seconds ago

Docker Package Repositories	Operational
Docker Hub Web	Operational
Docker Hub Registry	Operational
Docker Hub Automated Builds	Operational
Docker.com Web	Operational
Docker Docs	Operational
Docker Community Forums	Operational
Docker Support Site	Operational

Metrics

Today Week Month

DOCKER HUB API RESPONSE TIME 505.16MS

https://status.docker.com/

Metrics

Today Week Month

DOCKER HUB API RESPONSE TIME 500.71MS

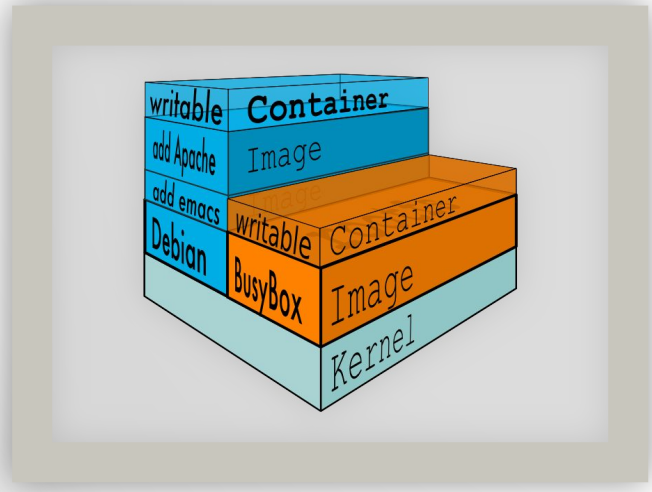
DOCKER HUB API UPTIME 100.00%

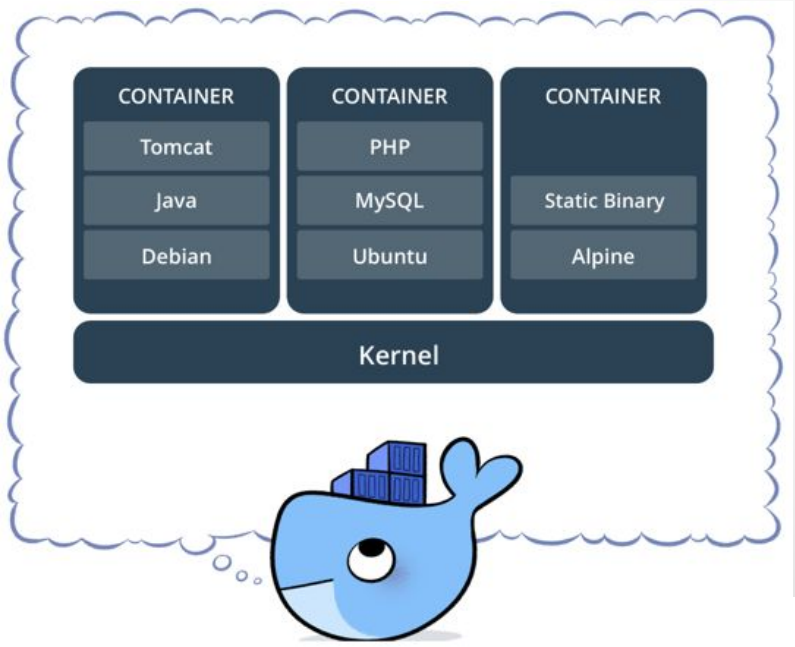
DOCKER REGISTRY HUB API RESPONSE TIME 504.03MS

DOCKER REGISTRY HUB API UPTIME 100.00%

Docker Container

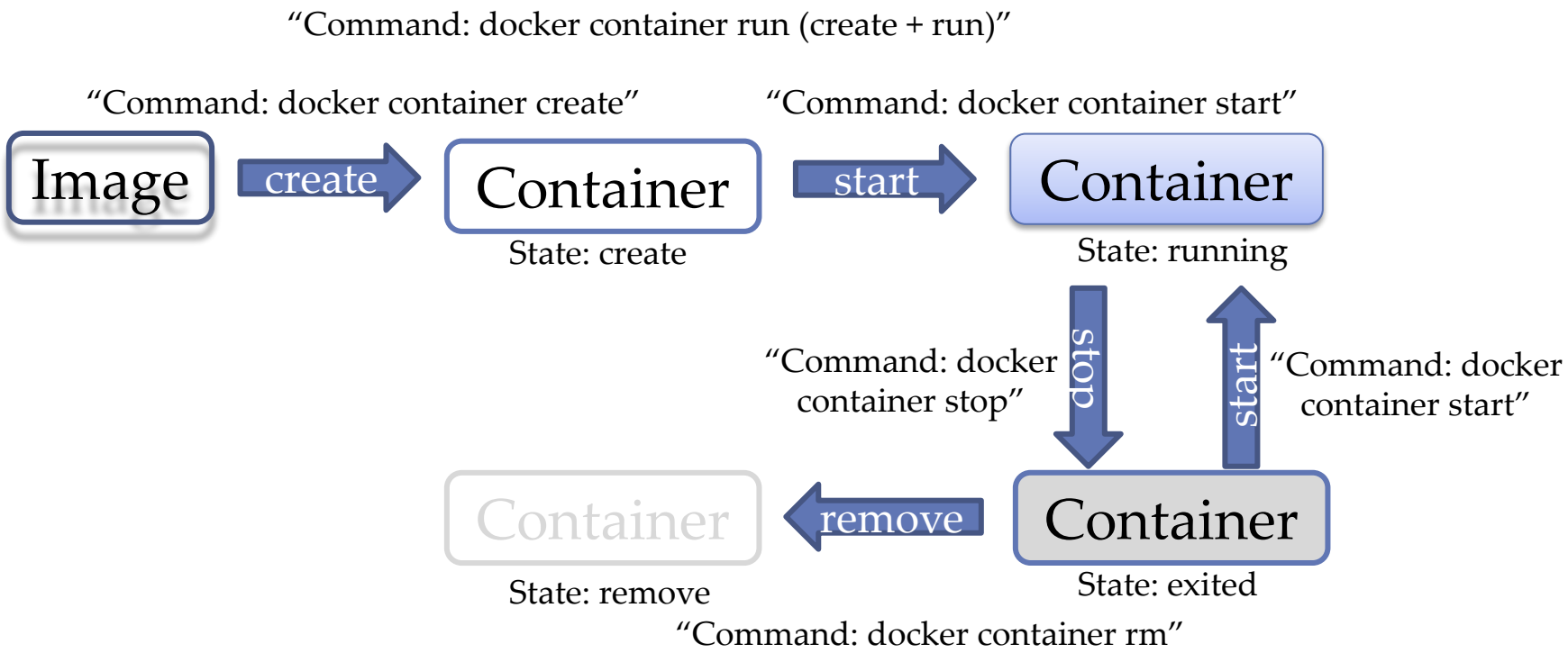
- Container is set of software layer and independence from image
- Container will provide the environment for developer/run application inside container
- When developer had been done. So we will export all application as image set and push to registry for deployment





Docker Container

- Container State:



Docker Container

- Container Different (Compare with Image):

docker container diff <container name>

```
ubuntu@ip-10-0-1-90:~$ docker container diff nodejs
C /nodejs
A /nodejs/test.js
C /root
A /root/.ash_history
ubuntu@ip-10-0-1-90:~$ docker container ls
CONTAINER ID   IMAGE                        COMMAND           CREATED          STATUS                    PORTS                                       NAMES
4b04b8ba10e0   labdocker/alpineweb:latest   "node hello.js"   48 seconds ago   Up 47 seconds (healthy)   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   nodejs
ubuntu@ip-10-0-1-90:~$
```

Docker Container

- Container Performance/Port:

docker container stats <container name>

docker container top <container name>

docker container port <container name>

```
CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
115bb4a5b92b        nodejs              0.00%               8.375MiB / 1.91GiB    0.43%               5.9kB / 2.3kB       36.9kB / 0B         6
^C
ubuntu@ip-10-0-1-58:~$ docker container top nodejs
UID                 PID                 PPID                C                   STIME               TTY                 TIME                CMD
root                3069                3043                0                   14:15               pts/0               00:00:00            node
hello.js
ubuntu@ip-10-0-1-58:~$ docker container port nodejs
3000/tcp -> 0.0.0.0:3000
ubuntu@ip-10-0-1-58:~$
```

Docker Container

- Container change name on-the-fly:

```
docker container rename <old name> <new name>
```

```
ubuntu@ip-10-0-1-90:~$ docker container ls
CONTAINER ID   IMAGE                        COMMAND           CREATED         STATUS                     PORTS                                       NAMES
4b04b8ba10e0   labdocker/alpineweb:latest   "node hello.js"   3 minutes ago   Up 49 seconds (healthy)    0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   nodejs
ubuntu@ip-10-0-1-90:~$ docker container rename nodejs nodejsnew
ubuntu@ip-10-0-1-90:~$ docker container ls
CONTAINER ID   IMAGE                        COMMAND           CREATED         STATUS                         PORTS                                       NAMES
4b04b8ba10e0   labdocker/alpineweb:latest   "node hello.js"   3 minutes ago   Up About a minute (healthy)    0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   nodejsnew
ubuntu@ip-10-0-1-90:~$
```

Docker Container

- Run container from image by command

```
docker container run <option> <image id/name> <command>
```

Ex: docker container run -i -t --rm -p 3000:3000 \
labdocker/alpineweb:latest node hello.js

- Run exec command for access existing container

```
docker container exec –it <container id/name> <command>
docker container attach <container id/name>
```

- Stop/Start container

```
docker container <start/stop> <container id/name>
```

- Delete container

```
docker container rm <containerid/name>
```

Workshop: Run Container

- <u>Interactive NODEJS</u>
- Run container with Interactive mode and set container name "nodejs" and map network 3000:3000

```
docker container run  -i -t --rm --name nodejs -p 3000:3000 \
labdocker/alpineweb:latest node hello.js
```

- Test open url: http://<ip address>:3000
- Check container status with command:

```
docker container ps -a
```

- Exist with Ctrl+C

Workshop: Run Container

- <u>DAEMON LINE BOT</u>

ธนาคารแห่งประเทศไทย

ข้อมูลทั่วไปของ API

ประเภทของ API :	REST
ประเภทของข้อมูล :	JSON
ความปลอดภัย :	PUBLIC

ตำแหน่งที่อยู่ของ API

https://iapi.bot.or.th/Stat/Stat-ReferenceRate/DAILY_REF_RATE_V1/

พารามิเตอร์ใน Header

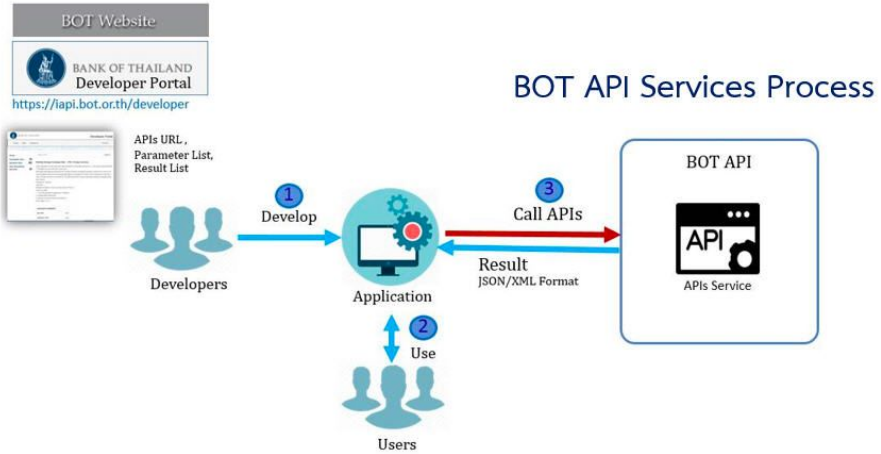
api-key : xk3h6ash-ahw5l3mch6hl3-2hhg6l4mng2346l34l6

ข้อมูลพารามิเตอร์สำหรับการใช้งาน API

ชื่อตัวแปร	ประเภท	คำอธิบาย
start_period	Datetime	วันที่เริ่มต้นงวดข้อมูล (YYYY-MM-DD) เช่น 2017-06-30
end_period	Datetime	วันที่สิ้นสุดงวดข้อมูล (YYYY-MM-DD) เช่น 2017-06-30

ข้อมูลพารามิเตอร์ที่จะได้รับ

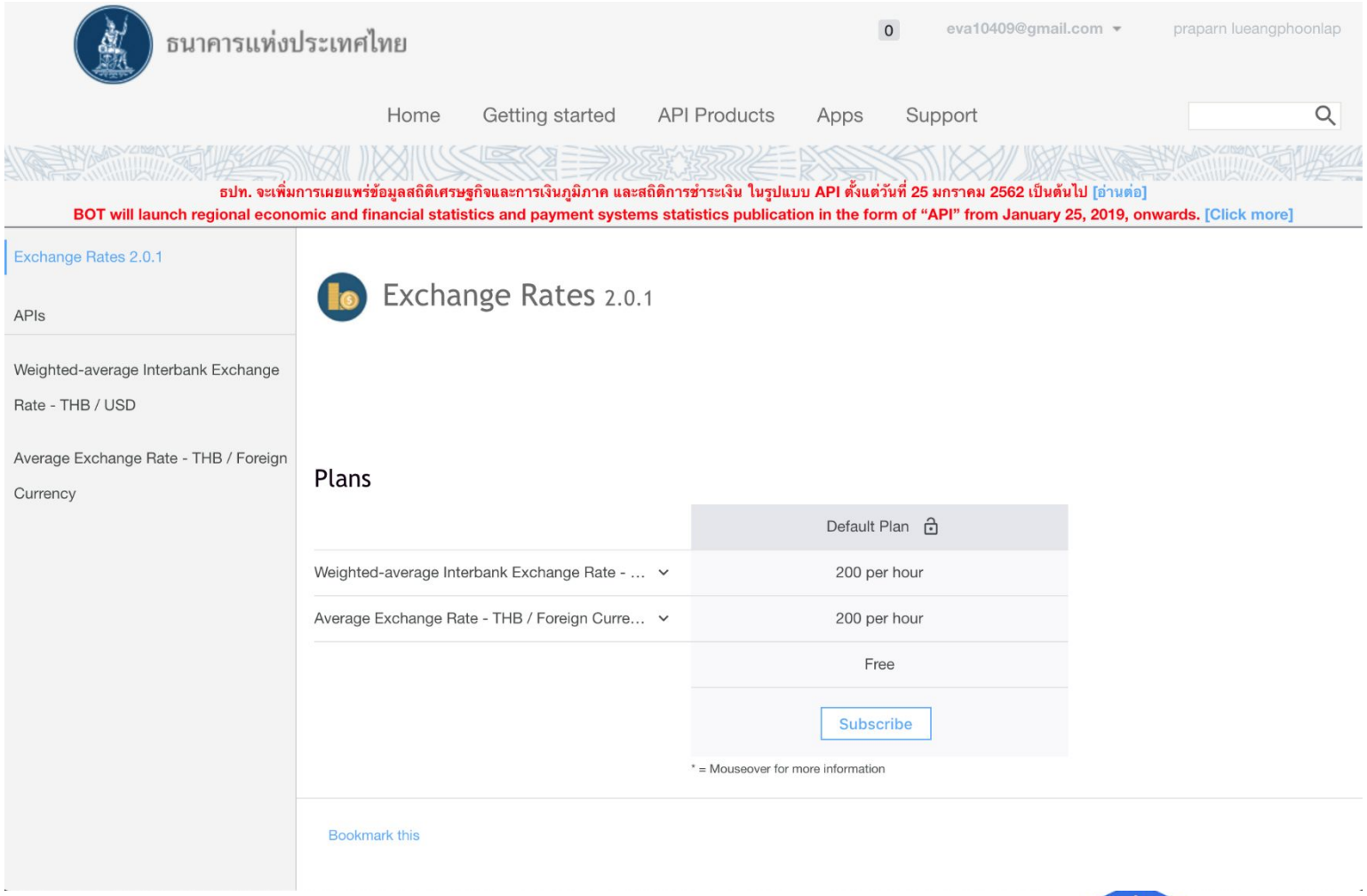
ชื่อ	ประเภท	คำอธิบาย
report_name_eng	String	ชื่อรายงาน ภาษาอังกฤษ
rereport_name_th	String	ชื่อรายงาน ภาษาไทย
report_uoq_name_eng	String	หน่วย ภาษาอังกฤษ
report_uoq_name_th	String	หน่วย ภาษาไทย
report_source_of_data	String	แหล่งที่มาของข้อมูล
report_remark_eng	String	หมายเหตุ รายงานภาษาอังกฤษ
report_remark_th	String	หมายเหตุ รายงานภาษาไทย
last_updated	String	วันที่แก้ไขข้อมูลล่าสุด
period	String	วันที่งวดข้อมูล
rate	String	อัตรา



kubernetes by Google

Workshop: Run Container

- Ex 2: DAEMON LINE BOT

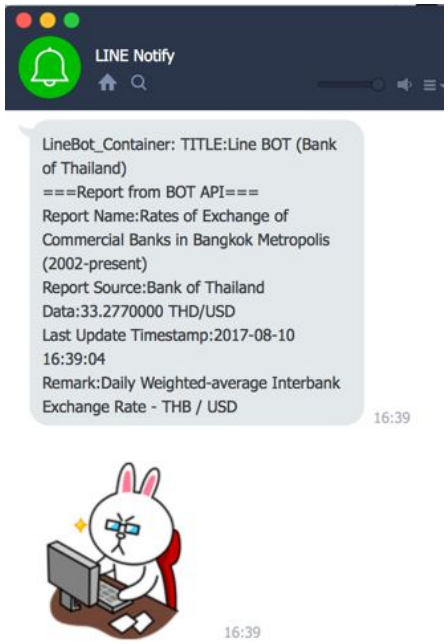


Workshop: Run Container

- DAEMON LINE BOT

```
docker@labdocker:~$ docker container run --rm --name linebot -e "TITLE=Line BOT (Bank of Thailand)" -e "TOKEN=EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX" labdocker/linenotify:bot_v1
null
{"statusCode":200,"body":"{\"status\":200,\"message\":\"ok\"}","headers":{"server":"nginx","date":"Sat, 20 Jan 2018 03:12:22 GMT","content-type":"application/json;charset=UTF-8","transfer-encoding":"chunked","connection":"keep-alive","keep-alive":"timeout=3","x-ratelimit-limit":"1000","x-ratelimit-imagelimit":"50","x-ratelimit-remaining":"998","x-ratelimit-imageremaining":"50","x-ratelimit-reset":"1516421345"},"request":{"uri":{"protocol":"https:","slashes":true,"auth":null,"host":"notify-api.line.me","port":443,"hostname":"notify-api.line.me","hash":null,"search":null,"query":null,"pathname":"/api/notify","path":"/api/notify","href":"https://notify-api.line.me/api/notify"},"method":"POST","headers":{"Content-Type":"application/x-www-form-urlencoded","authorization":"Bearer EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX","content-length":466}}}
"{\"status\":200,\"message\":\"ok\"}"
docker@labdocker:~$
```

```
const request = require('request');
const qs = require('querystring');
const TITLE=process.env.TITLE;
const KEY =process.env.KEY; //Input API-KEY to Header
//const KEY="U9G1L457H6DCugT7VmBaEacbHV9RX0PyS005cYaGsm";
const URL =process.env.URL; //Input URL for Request
//const URL="https://iapi.bot.or.th/Stat/Stat-ReferenceRate/DAILY_REF_RATE_V1/?";
//const TOKEN = 'EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX';
const TOKEN=process.env.TOKEN; //Input TOKEN LINE
//Set Date
var dateFormat = require('dateformat');
var daynow = new Date(); // Today!
daynow.setDate(daynow.getDate() - 1); // Yesterday!
var daynow=dateFormat(daynow, "yyyy-mm-dd");
var DAYSTARTPERIOD=daynow;
var DAYENDPERIOD=daynow;
//Set Date
var stickerPkg=2; //stickerPackageId
var stickerId=161; //stickerId
var Message="";
//Setup QueryString
var querystring = qs.stringify({
    start_period: DAYSTARTPERIOD,
    end_period: DAYENDPERIOD
});
//Setup QueryString
function callback(error, response, body) {
  //console.log(URL+querystring);
  //console.log('Response Code: '+ response.statusCode);
  //console.log(body)
  if (!error && response.statusCode == 200) {
    var info = JSON.parse(body);
```



Workshop: Run Container

- <u>DAEMON LINE BOT</u>
- Run container with daemon mode as set name to "linebot" for retrieve data from API of BOT and show LINE Notify with Token Key

```
docker container run -it --rm --name linebot \
-e TITLE="Line BOT (Bank of Thailand)" \
 -e "TOKEN=<LINE TOKEN>"
labdocker/linenotify:bot_v2
```

Workshop: Run Container

- <u>Detach Mode NODEJS</u>
- Run container with detach mode (daemon) and set container name "nodejs" and map network 3000:3000

```
docker container run -d -t --name nodejs -p 3000:3000 \
labdocker/alpineweb:latest node hello.js
```

- Test open url: http://<ip address>:3000
- Check container status with command:

```
docker container ps -a
```

- Test access to container with container

```
docker container exec –i –t nodejs sh
```

Workshop: Run Container

- Detach Mode NODEJS
- Stop container with command

```
docker container stop nodejs
```

```
docker container start nodejs
```

- Check container status with command:

```
docker container ls -a
```

- Test access to container with container

```
docker container exec –i –t nodejs sh
```

```
docker container attach nodejs
```

Workshop: Run Container

- <u>Detach Mode NODEJS</u>
- Verify process inside container by command:

```
docker container top nodejs
```

```
docker container stats nodejs
```

- Verify difference between container and image by command:

```
docker container diff nodejs
```

- Test port mapping on container

```
docker container port nodejs
```

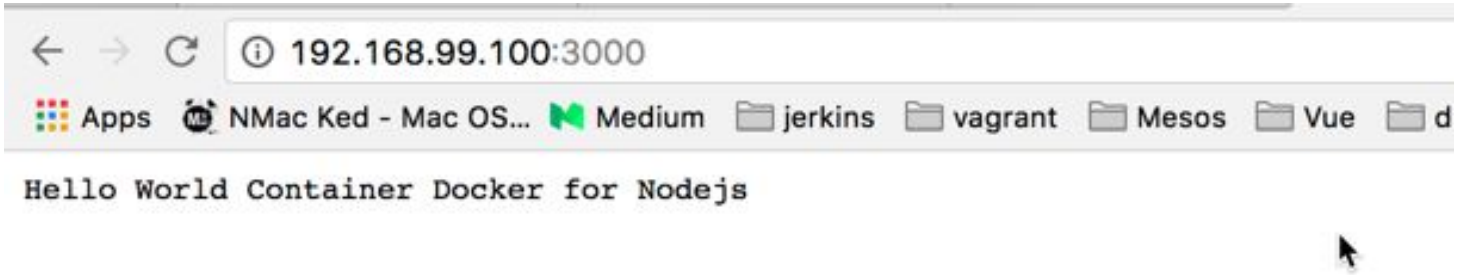
- Online change container name

```
docker container rename nodejs nodejsnew
```

Workshop: Run Container

- Detach Mode NODEJS

```
1  var http = require('http');
2
3  http.createServer(function (req, res) {
4    res.writeHead(200, {'Content-Type': 'text/plain'});
5    res.end('Hello World Container Docker for Nodejs\n');
6  }).listen(3000, '0.0.0.0');
7
8  console.log('Server running at http://0.0.0.0:3000/');
```



Workshop: Run Container

- <u>Detach Mode Python</u>
- Create container as detach (daemon) as set name to "python" and map network 5000:5000

```
docker create run -t --name python -p 5000:5000 \
labdocker/cluster:webservicelite
```

- Test Run

```
docker container ls –a
docker container start python
```

- Test one url: http://<ip address>:5000
- Shell to access container as command below

```
docker container exec –i –t python sh
```

```
docker container attach python
```

Workshop: Run Container

- Detach Mode Python

```python
from flask import Flask
import os
import time
app = Flask(__name__)

@app.route('/')
def hello():
    return '<H1> 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)
```

CPU, Memory and I/O

CPU

- Default container resource of all cpu resource and share cpu time all all container with same ratio

- Config share time (default: 1024)

```
--cpu-shares, -c =“<0 (default): Ratio>”
```

- Config specify core cpu set

```
--cpuset-cpus=“1, 3"
```

- Define cpu quota & period (default: 100 ms)(CFS scheduler)

```
--cpu-period=100000 --cpu-quota=500000
```

CPU

- Set NUMA (non-uniform memory access) architecture

```
--cpuset-mems="1,3"
```

- Define cpu real-time runtime (ms) (nice system call)

```
--cap-add=sys_nice --cpu-rt-runtime=<value>
```

- Define cpu run-time priority (0-99) (nice system call)

```
--cap-add=sys_nice --ulimit rtprio=xx
```

NVIDIA GPU

- Docker can operate with GPU of NVIDIA and require to install nvidia-container-runtime as step below

```
apt-get install nvidia-container-runtime
which nvidia-container-runtime-hook
```

- Check gpu available for use in system by command

```
docker run -it --rm --gpus all ubuntu nvidia-smi
```

- Run docker container with gpu:

```
docker run -it --rm --gpus device=xxxx ubuntu nvidia-smi
```

Ref:https://github.com/NVIDIA/nvidia-docker

Workshop: CPU Configure

- Download cAdvisor

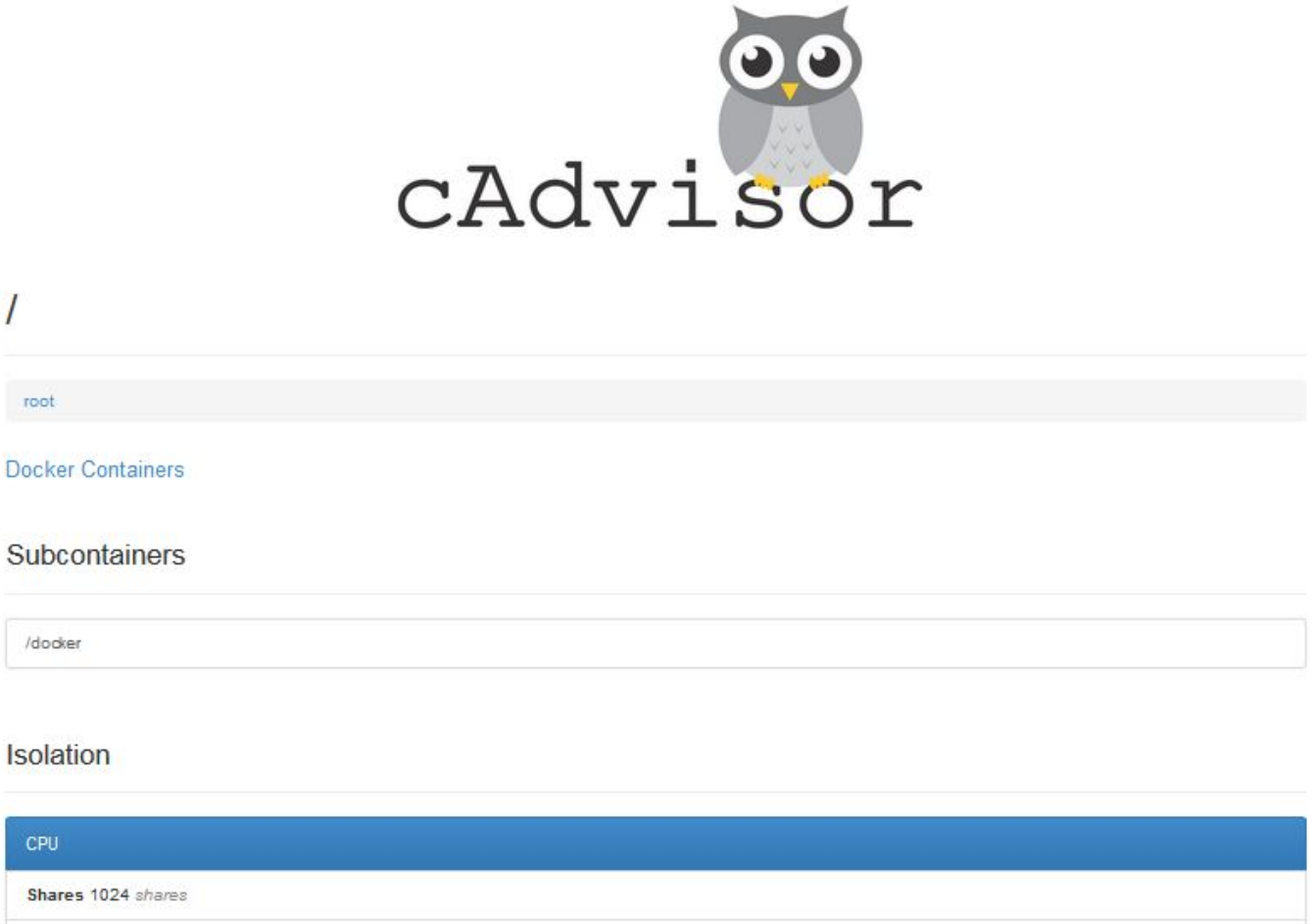
```
docker image pull labdocker/cadvisor:0.40.0
```

- Run container cAdvisor by command

```
docker container run \
    --mount type=bind,source=/var/run,target=/var/run \
    --mount type=bind,source=/sys,target=/sys,readonly \
    --mount
type=bind,source=/var/lib/docker,target=/var/lib/docker,readonly \
    --publish=8080:8080 \
    --detach=true \
    --name=cadvisor \
    labdocker/cadvisor:0.40.0
```

Workshop: CPU Configure

- cAdvisor



Workshop: CPU Configure

- Download busybox for test

```
docker image pull labdocker/busybox:latest
```

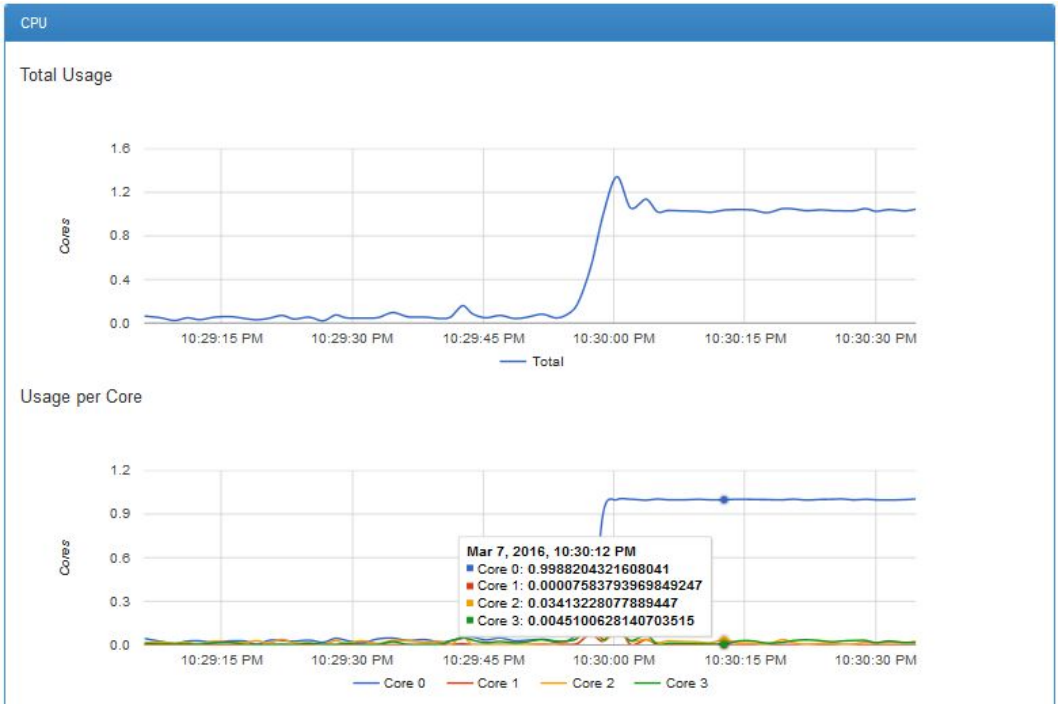
- Scenario 1 (Single CPU: 70/30)

```
docker container run -d \
--name='Share_30' \
--cpuset-cpus=0 \
--cpu-shares=30 \
labdocker/busybox:latest md5sum /dev/urandom
```

```
docker container run -d \
--name='Share_70' \
--cpuset-cpus=0 \
--cpu-shares=70 \
labdocker/busybox:latest md5sum /dev/urandom
```

Workshop: CPU Configure

- Result




```
CONTAINER           CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
01109e1c7b0f        70.65%              76KiB / 995.4MiB      0.01%               648B / 0B           0B / 0B             1
909882f27a59        30.26%              40KiB / 995.4MiB      0.00%               648B / 0B           0B / 0B             1
e247e2e8240a        0.05%               16.86MiB / 995.4MiB   1.69%               718kB / 39.4MB      17.6MB / 0B         11
^C
docker@labdocker:~$ docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS
           NAMES
01109e1c7b0f        labdocker/busybox:latest    "md5sum /dev/urandom"    2 minutes ago       Up 2 minutes
          Share_70
909882f27a59        labdocker/busybox:latest    "md5sum /dev/urandom"    3 minutes ago       Up 3 minutes
          Share_30
e247e2e8240a        labdocker/cadvisor:latest   "/usr/bin/cadvisor..."   4 minutes ago       Up 4 minutes        0.0.0.0:8080->
8080/tcp   cadvisor
docker@labdocker:~$
```

Quiz ?

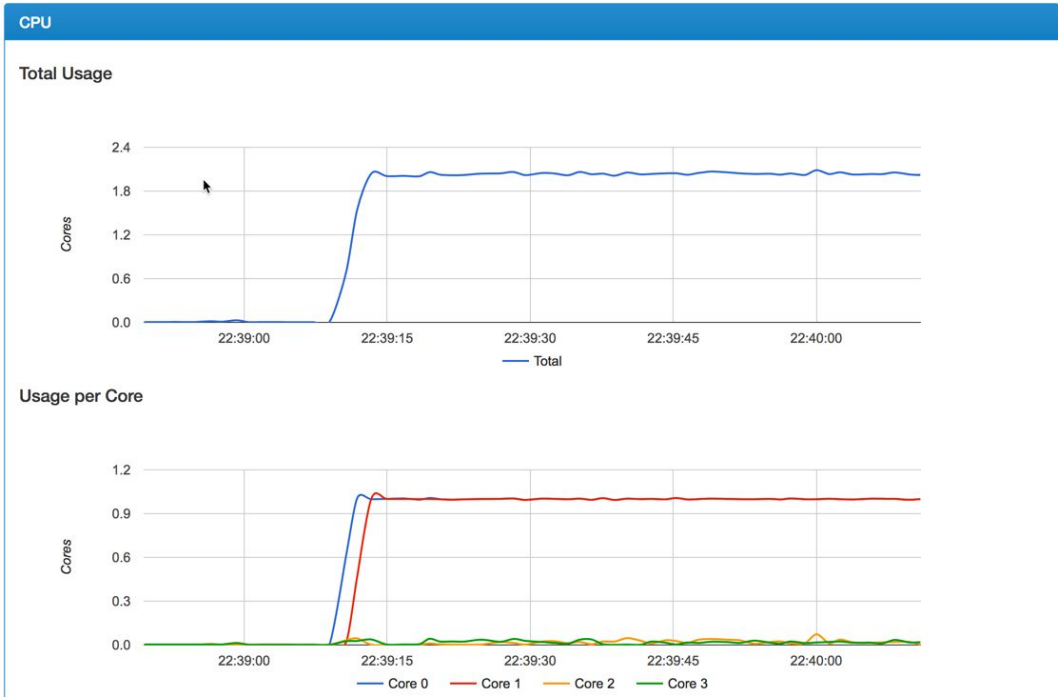
- Scenario (Multiple CPU: 70/30)

```
docker container run -d \
--name='Share_30' \
--cpuset-cpus=0 \
--cpu-shares=30 \
labdocker/busybox:latest md5sum /dev/urandom
```

```
docker container run -d \
--name='Share_70' \
--cpuset-cpus=1 \
--cpu-shares=70 \
labdocker/busybox:latest md5sum /dev/urandom
```

Quiz ?

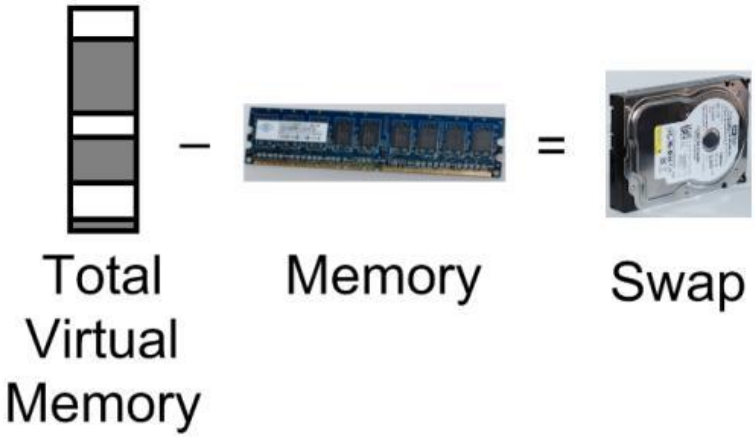
- Result

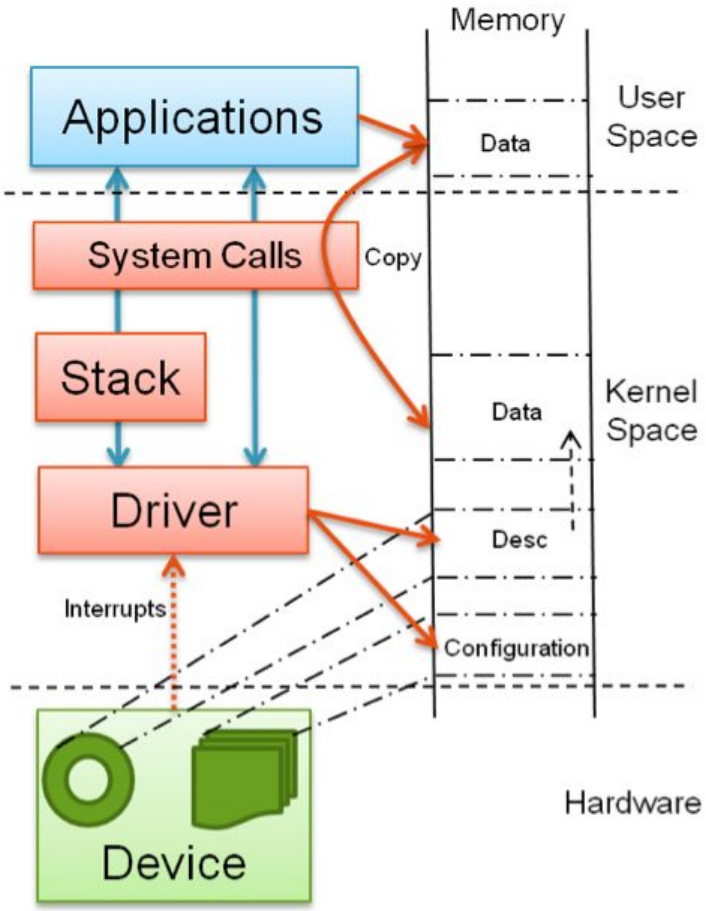



```
CONTAINER           CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
02d0b2973cff        99.93%              88KiB / 995.4MiB      0.01%               648B / 0B           0B / 0B             1
b1aebf2a3d71        99.88%              84KiB / 995.4MiB      0.01%               648B / 0B           0B / 0B             1
e247e2e8240a        2.02%               19.34MiB / 995.4MiB   1.94%               1.07MB / 66.7MB     17.6MB / 0B         11
^C
docker@labdocker:~$ docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS
           NAMES
02d0b2973cff        labdocker/busybox:latest    "md5sum /dev/urandom"    41 seconds ago      Up 40 seconds
           Share_70
b1aebf2a3d71        labdocker/busybox:latest    "md5sum /dev/urandom"    42 seconds ago      Up 42 seconds
           Share_30
e247e2e8240a        labdocker/cadvisor:latest   "/usr/bin/cadvisor..."   6 minutes ago       Up 6 minutes        0.0.0.0:8080->
8080/tcp   cadvisor
docker@labdocker:~$
```

Memory

- Default container will show all of memory resource during runtime and can utilization





Memory

- Memory configuration
- memory limit and reservation (default: unlimit) (B:byte, K:kilobyte, M:megabyte, G:gigabyte)

```
--memory, -m =10G --memory-reservation= 1G
```

- memory swap (default: unlimit)

```
-- memory-swap= -1
```

- kernel memory

```
-- kernel-memory= 500M
```

- memory swappiness (kernel)(0=no swap)

```
-- memory-swappiness =0
```

- disable oom kill

```
--oom-kill-disable =0
```

Memory

- Memory on production system
- Memory-swappiness = 0 (no swappiness for kernel memory)
- Define memory swap (--memory-swap)x2 of physical
- Monitor memory footprint of application system and define “--memory-reservation”
- Capacity testing for define maximum memory (+30%) and define “--memory“ (Limit)
- Critical system should be “--memory-reservation” with “--memory“ (Limit)

I/O

- By default all container have priority I/O as same (500 weight)

--blkio-weight 300, --blkio-weight-device “/s01/tdb:500”

- Define weight for I/O (Direct IO)

--device-read-bps /s01/tdb:1mb

--device-read-iops /s01/tdb:20000

- Limit I/O with bps (kb: kilobyte, mb: megabyte, gb: gigabyte) / iops

--device-write-bps /s01/tdb:1mb

--device-write-iops /s01/tdb:20000

I/O

- I/O configure in production system
- weight I/O should be define for application with direct i/o (slow than cache)
- Monitor I/O utilization by command: (IOSTAT –xtc)

```
$ iostat -xtc
                    extended device statistics                          tty         cpu
device       r/s    w/s   kr/s   kw/s wait actv  svc_t  %w  %b tin tout  us sy wt id
fd0          0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0   0    0   0  0  0 100
sd0          0.0    0.0    0.4    0.4  0.0  0.0   49.5   0   0
sd6          0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0
nfs1         0.0    0.0    0.0    0.0  0.0  0.0    0.0   0   0
nfs49        0.0    0.0    0.0    0.0  0.0  0.0   15.1   0   0
nfs53        0.0    0.0    0.4    0.0  0.0  0.0   24.5   0   0
nfs54        0.0    0.0    0.0    0.0  0.0  0.0    6.3   0   0
nfs55        0.0    0.0    0.0    0.0  0.0  0.0    4.9   0   0
```

- Web server / Application server ⇒ Hi IOPS / Low Transfer
- Database server ⇒ Low IOPS / Hi Transfer

I/O

- Normal disk iops

Disk Speed	IOPS
15,000	175
10,000	125
7200	75
5400	50

RAID	Write Penalty
0	1
1	2
5	4
6	6
DP	2
10	2

- Raw disk performance: disk iops x unit of disk
  - **Ex:** SAS 128GB (15K) x 6 units => 175 x 6 = 1060 iops
- Raid disk performance:
  - ((raw iops x %write)/penalty)+(raw x %read)
  - Normal situation (write 30%, read 70%)
  - **Ex:** Raid 5 ((1060 x .3)/4)+(1060x.7) □ 821.6 iops
  - **Ex**: Raid 1 ((1060 x .3)/2)+(1060x.7) □ 901 iops

Network

Network

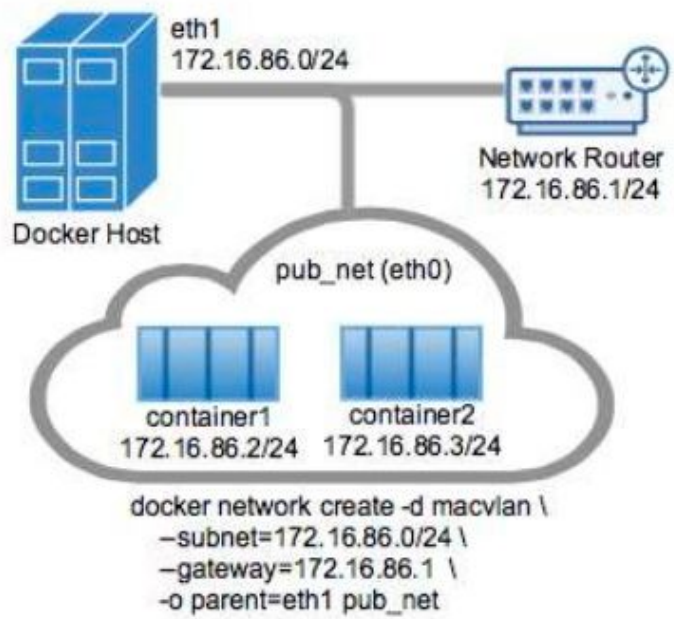
- Software define network by design (virtual switch) (Support IPv4/6)
- Docker will have 3 type of default network

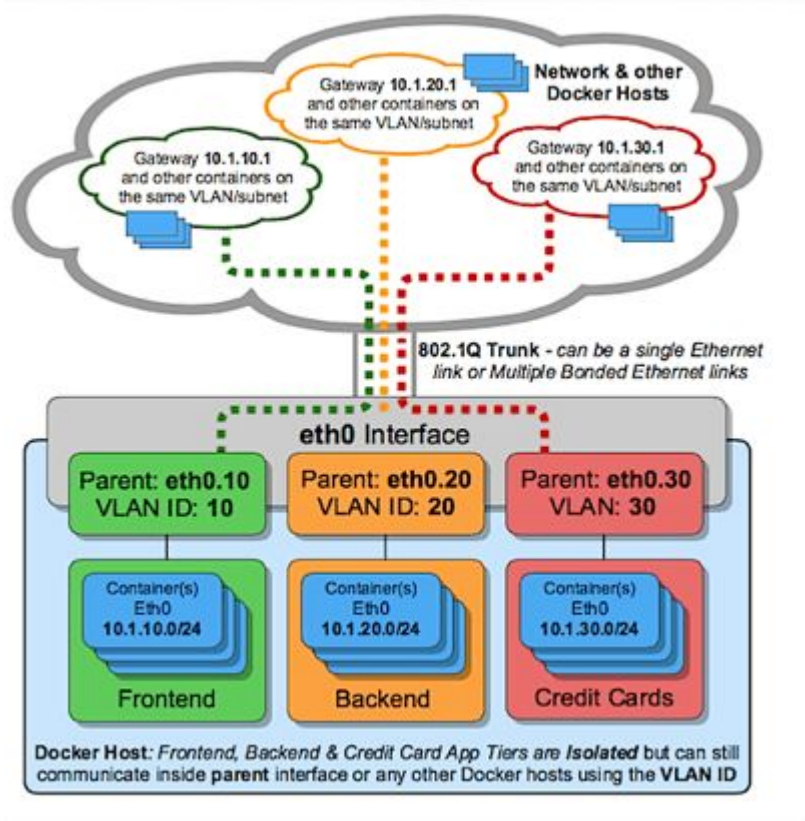
```
docker@labdocker:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
0dc1bbf00498        bridge              bridge              local
91d6dd4056a9        host                host                local
fa786a00adda        none                null                local
docker@labdocker:~$
```

- **Bridge/User Defined**: default network for access to outside via virtual switch "docker0" on each container have their network stack
- **none:** network loopback (127.0.0.1)
- **host**: network host for use network card of host to operate

Network

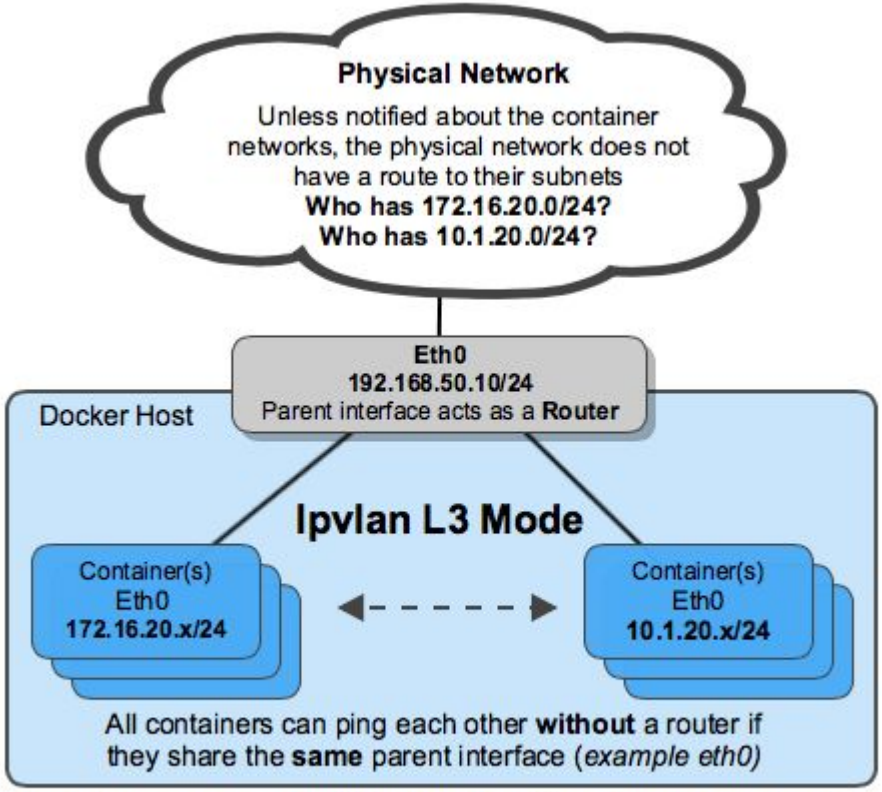
- Additional network type
- **MACVLAN/IPVLAN (L2)**





Network

- Additional network ty
- **IPVLAN (L3 Mode)**



Containers can be on different subnets and reach each other

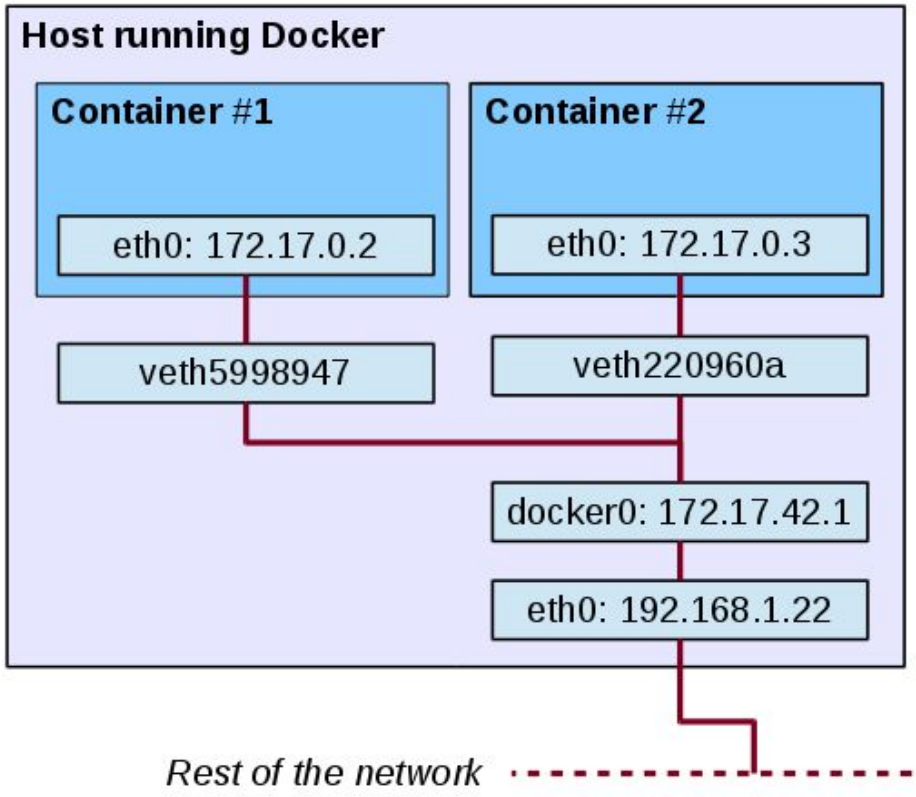
- **Overlay (Swarm Mode)**
- **3rd Party Network Plugin (Swarm Mode)**

Network

- **Summary**
  - **Bridge/User Defined Network**: best when you need multiple containers to communicate on the same Docker host.
  - **Host networks:** best when the network stack should not be isolated from the Docker host, but you want other aspects of the container to be isolated.
  - **Overlay networks:** best when you need containers running on different Docker hosts to communicate, or when multiple applications work together using swarm services.
  - **Macvlan networks:** best when you are migrating from a VM setup or need your containers to look like physical hosts on your network, each with a unique MAC address
  - **Third-party network:** allow you to integrate Docker with specialized network stacks.

Network

- Bridge (default) Running container will assign ip address and connect to bridge network automatic



Network

- docker network inspect bridge

192.168.99.100 - PuTTY

```
docker@labdocker:~$ docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "5c03ed5297b82ab55b592d36324be256190b48267780a12bb603da9fd0c13887",
        "Scope": "local",
        "Driver": "bridge",
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16"
                }
            ]
        },
        "Containers": {
            "303c0f349f5ae46873f0dfb1440cae34a30d7e58e187c448f95842c475fb770f": {
                "Name": "cadvisor",
                "EndpointID": "5aa767acbcaa4d64095778b1daca4823b92dd497fa6f4aaa436b
9032e475fcc2",
                "MacAddress": "02:42:ac:11:00:02",
                "IPv4Address": "172.17.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        }
    }
]
docker@labdocker:~$
```

Network

- Option of Network

--dns= x.x.x.x (Default --name,--net-alias,--link also dns internal docker)
--network="<bridge/none/host/custom>"
--network-alias = "xxxx"
--add-host="xxxx"
--mac-address="xx:xx:xx:xx"
--ip="x.x.x.x"
--ipv6="xx:xx:xx:xx"
–link-local-ip="x.x.x.x"
-p, --publish = 9999:9999
-P, --publish-all Auto map network

```
docker container run  -i -t --rm --name nodejs -p 3000:3000 \
labdocker/alpineweb:latest node nodejs/hello.js
```

Network

- Create custom switch (Recommend for Production)

```
docker network create --driver  <default/null/host> name
```

```
Ex: docker network create --driver  default netweb
```

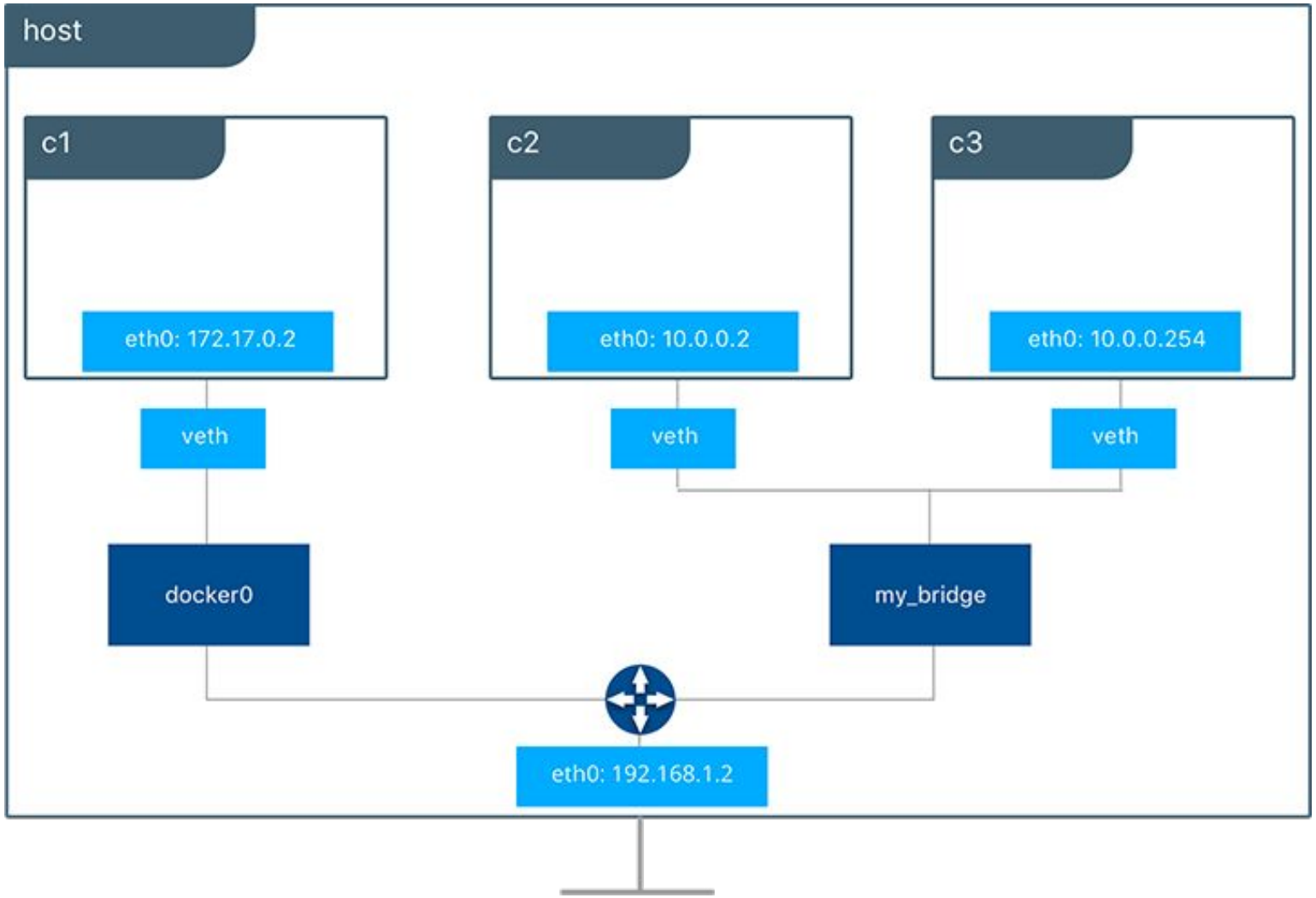
```
Ex: docker network rm netweb
```

- Option for custom switch
  - --attachable = Define to allow to attach
  - --aux-address = Define ip address (v4/v6) for network driver
  - --config-from= copy configuration from other virtual switch
  - --driver, -d =type virtual switch (bridge, none, macvlan etc)
  - --ingress = docker ee with service routeing-mesh
  - --internal = use only in docker endpoint
  - --scope = define scope of network (host,cluster)

Network

- Option for custom switch
  - --subnet = xx (define subnet in virtual switch (Ex: 192.168.100.0/24))
  - --ip-range = xx (define ip address range for container) (Ex: 192.168.100.128/25)
  - --gateway = xx (define gateway) (Ex: 192.168.100.5)
  - --opt = custom option
    - --opt="com.docker.network.mtu"="1500"
    - --opt="com.docker.network.bridge.host_binding_ipv4"=x.x.x.x

Network


host
c1
eth0: 172.17.0.2
c2
eth0: 10.0.0.2
c3
eth0: 10.0.0.254
veth
veth
veth
docker0
my_bridge
eth0: 192.168.1.2

Network

- Add network to container

```
docker network connect <network > <container >
```

```
Ex: docker network connect webnet web1
```

- Remove network from container

```
docker network disconnect <network > <container >
```

```
Ex: docker network disconnect webnet web1
```

Network

- Link container (add on /etc/hosts) (Legacy)

```
docker –dt --name web1 labdocker/alpineweb sh
```

```
docker –dt --name web2 --link web1:webmaster \
labdocker/alpineweb sh
```

Legacy container links

Estimated reading time: 14 minutes

> Warning: The `--link` flag is a legacy feature of Docker. It may eventually be removed. Unless you absolutely need to continue using it, we recommend that you use user-defined networks to facilitate communication between two containers instead of using `--link`. One feature that user-defined networks do not support that you can do with `--link` is sharing environmental variables between containers. However, you can use other mechanisms such as volumes to share environment variables between containers in a more controlled way.

- DNS resolve on docker network (Recommend for Production)

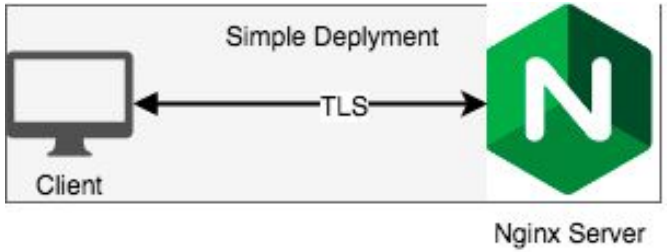
Workshop: Network Configure

- Part 1: Reverse Proxy Network (DNS)
- Purpose: Deploy nodejs webserver with solution reverse proxy by nginx (load balance / reverse proxy)

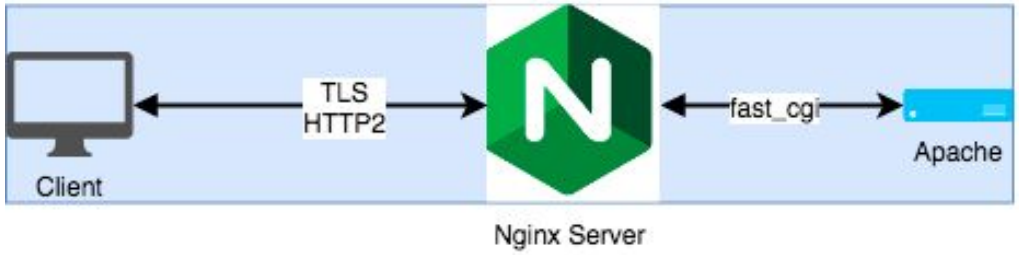
- **public network**: Allow external access via nginx server (reverse proxy)
  - IP Address: 192.168.100.0/24
  - Range Address: 192.168.100.128 – 192.168.100.256
  - MTU: 1500
- **private network:** Connection for nginx access inside container
  - IP Address: 192.168.101.0/24
  - Range Address: 192.168.101.128 – 192.168.101.256
  - MTU: 9000

Workshop: Network Configure

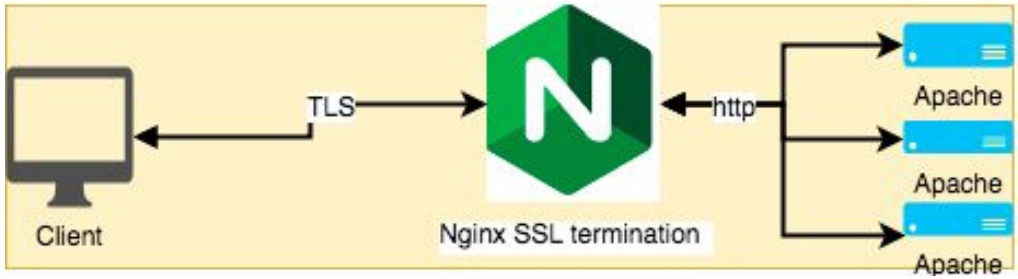
Workshop: Network Configure



Nginx Server A
Plain Deployment

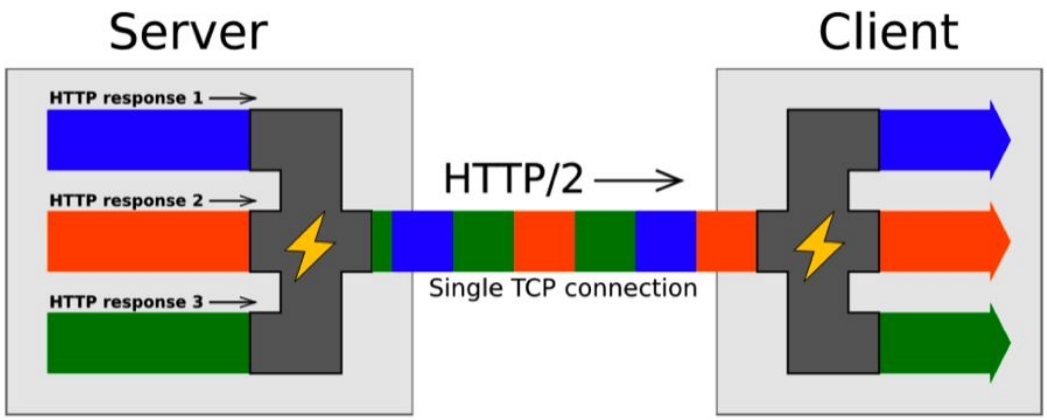


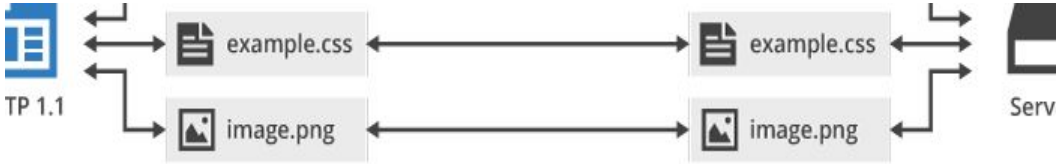
Nginx Reverse Proxy

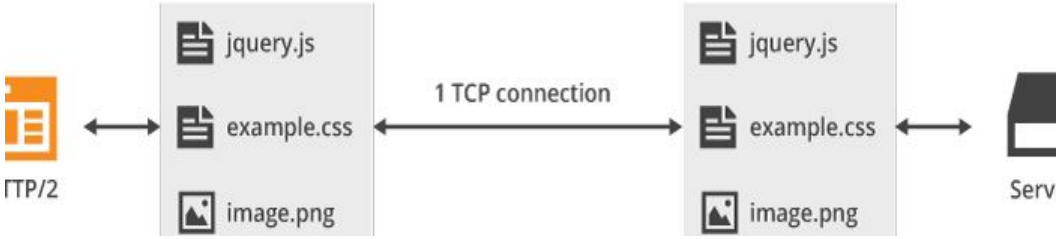


Nginx SSL Offloading

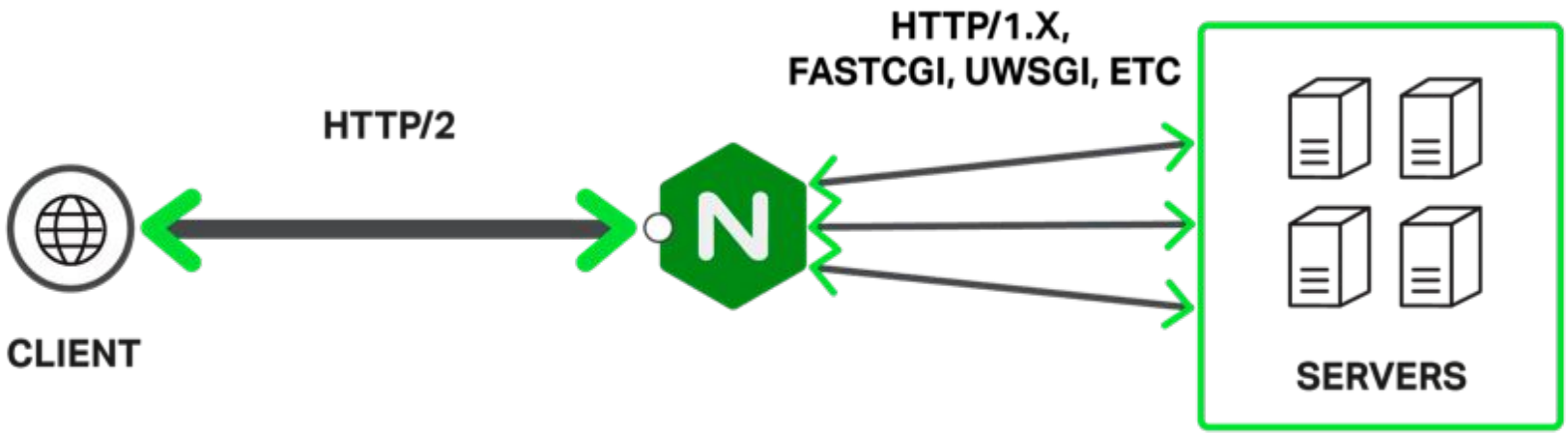
Workshop: Network Configure



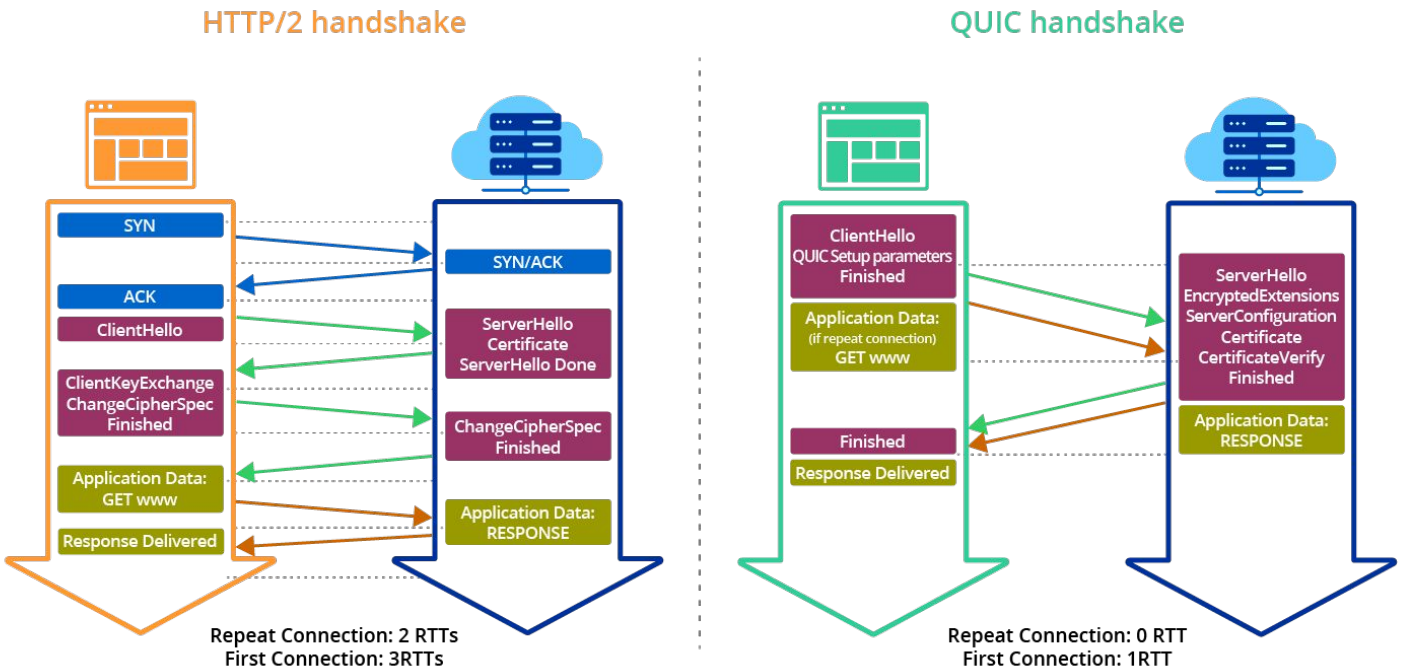


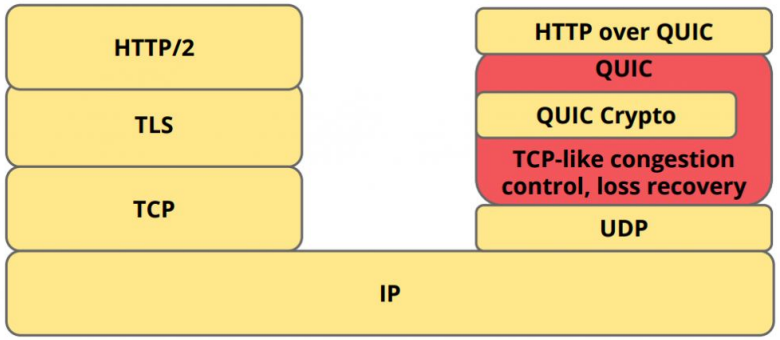


Workshop: Network Configure



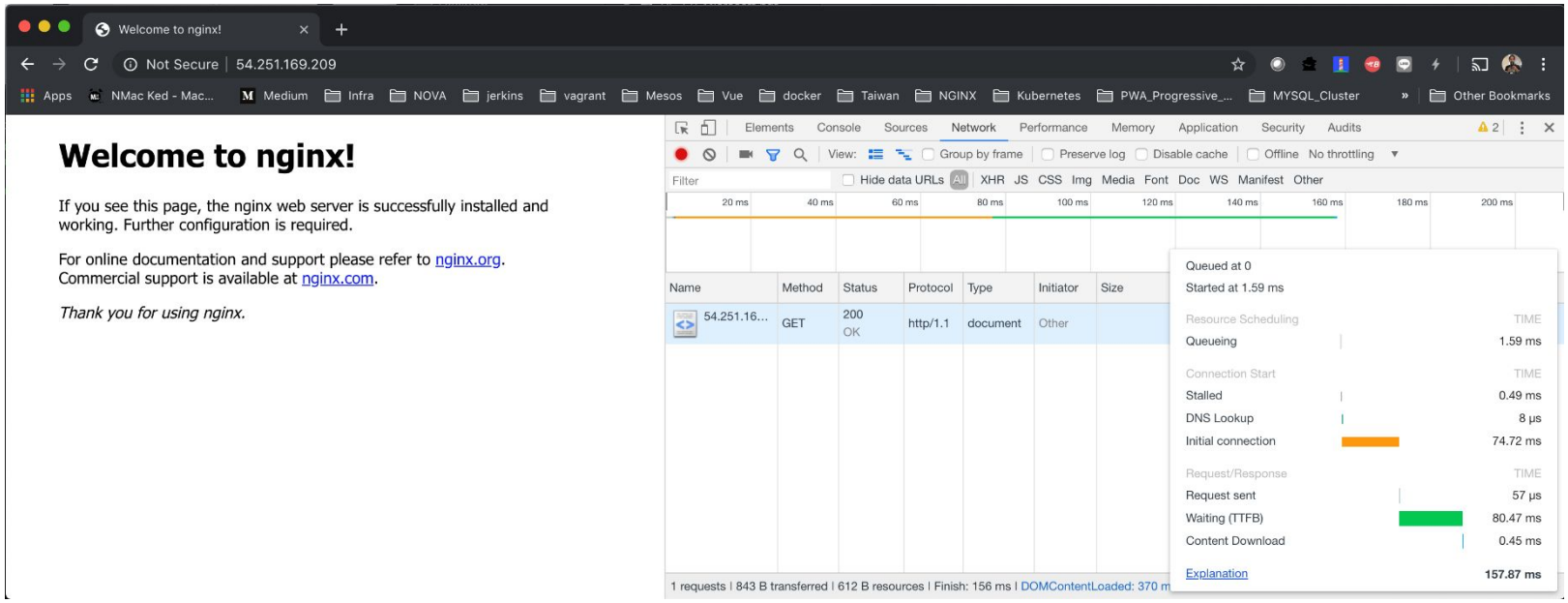
Workshop: Network Configure





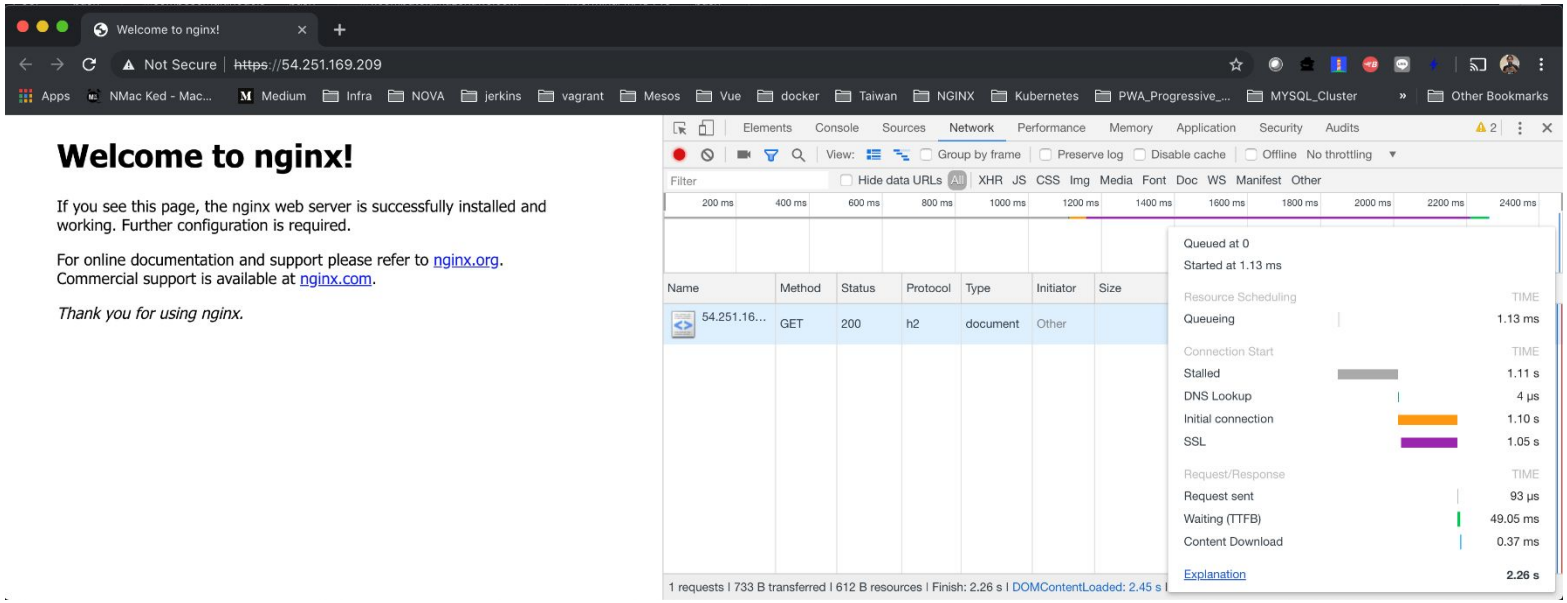
Workshop: Network Configure

- Normal HTTP/1.1

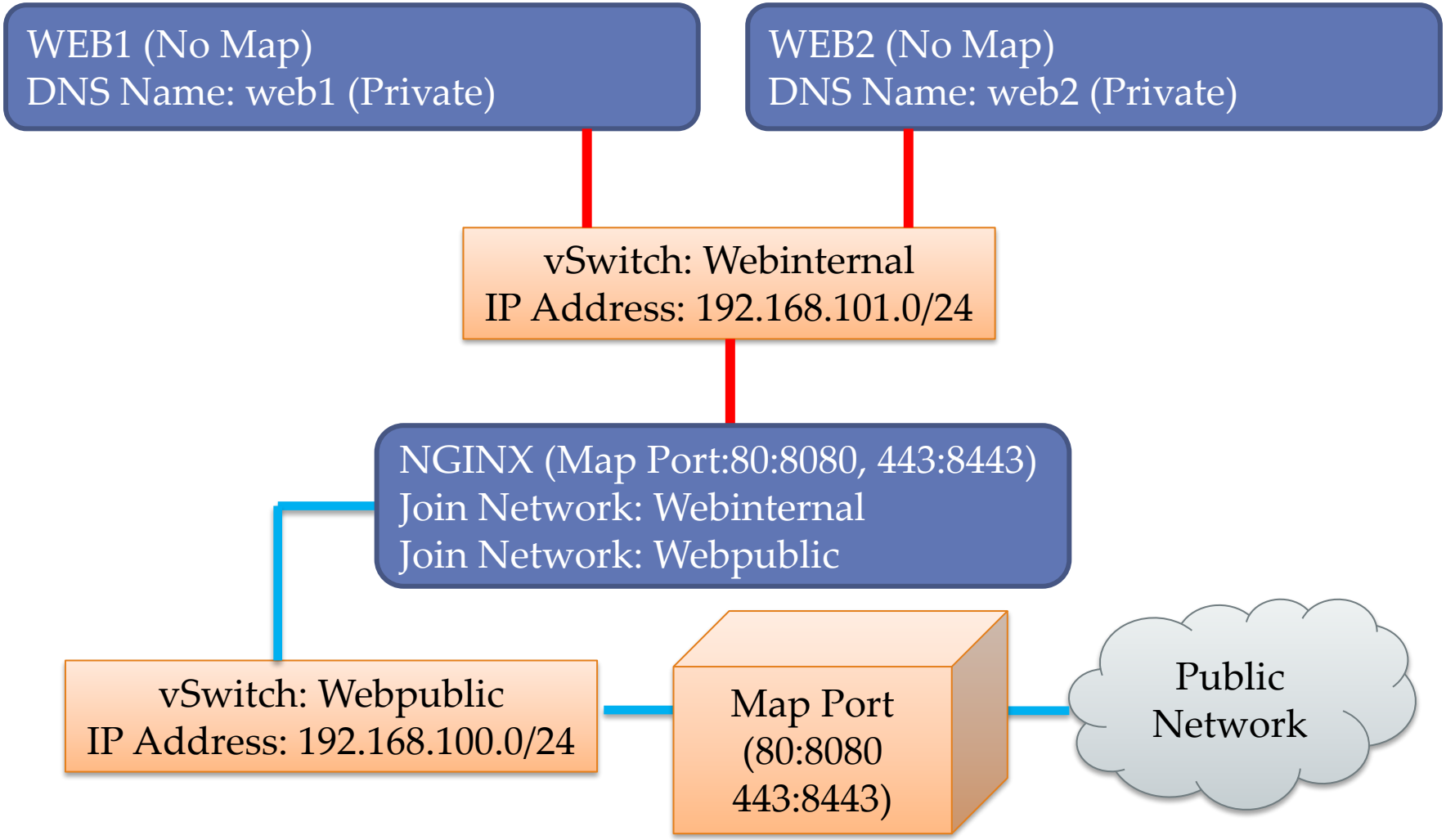


Workshop: Network Configure

- HTTP/2 HTTPS (Old word: SPDY)



Workshop: Network Configure



Workshop: Network Configure

- Create Switch Private / Public Network
  - docker network create --driver bridge \
  --subnet=192.168.100.0/24 --ip-range=192.168.100.128/25 \
  --gateway=192.168.100.5
  --opt="com.docker.network.mtu"="1500" webpublic

  - docker network create --driver bridge \
  --subnet=192.168.101.0/24 --ip-range=192.168.101.128/25 \
  --gateway=192.168.101.5
  --opt="com.docker.network.mtu"="9000" webinternal

Workshop: Network Configure

- Create WebServer and Nginx
  - **Web1**
  - docker run -dt --name web1 --net webinternal \
    --net-alias web1 labdocker/alpineweb:web1 node hello.js

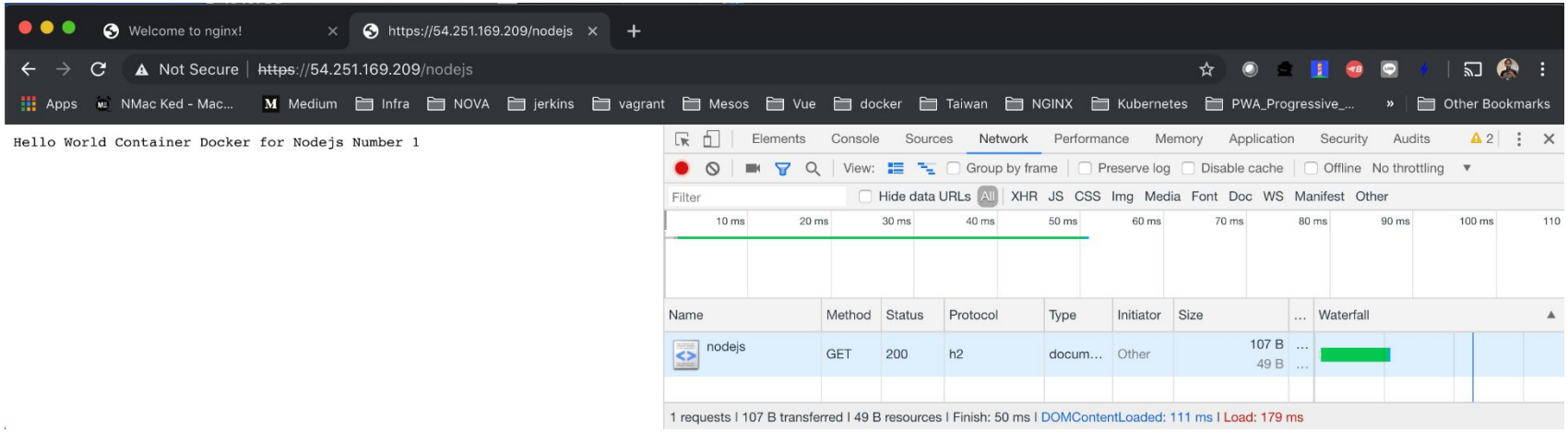
  - **Web2**
  - docker run -dt --name web2 --net webinternal \
    --net-alias web2 labdocker/alpineweb:web2 node hello.js

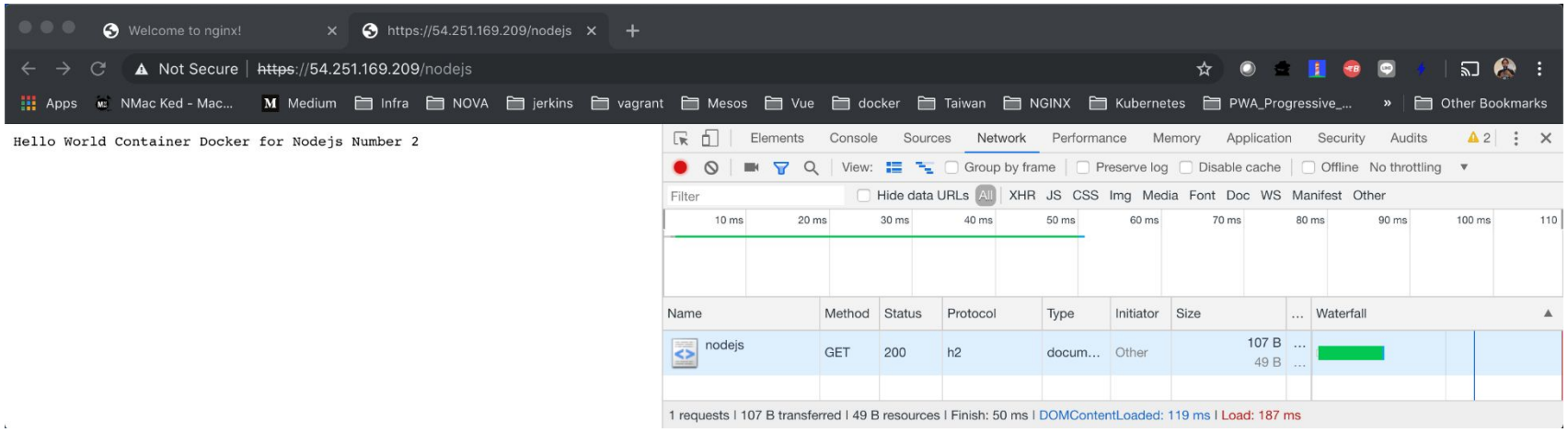
  - **NGINX**
  - docker run -dt --name nginx --net webinternal \
    -p 80:8080 –p 443:8443 labdocker/nginx:labnetworkhttp2

  docker network connect webpublic nginx

Workshop: Network Configure

- url: http://<Public IP>/nodejs





Workshop: Network Configure

- curl result:

```
...ntu@ip-10-0-1-55: ~ — -bash ...   ...composemultinodejs — -bash
ubuntu@ip-10-0-1-55:~/temp$ docker container exec -it nginx sh
/usr/sbin # curl http://web1:3000
Hello World Container Docker for Nodejs Number 1
/usr/sbin # curl http://web1:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 02:50:39 GMT
Connection: keep-alive

/usr/sbin # curl http://web2:3000
Hello World Container Docker for Nodejs Number 2
/usr/sbin # curl http://web2:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 02:50:40 GMT
Connection: keep-alive

/usr/sbin #
```

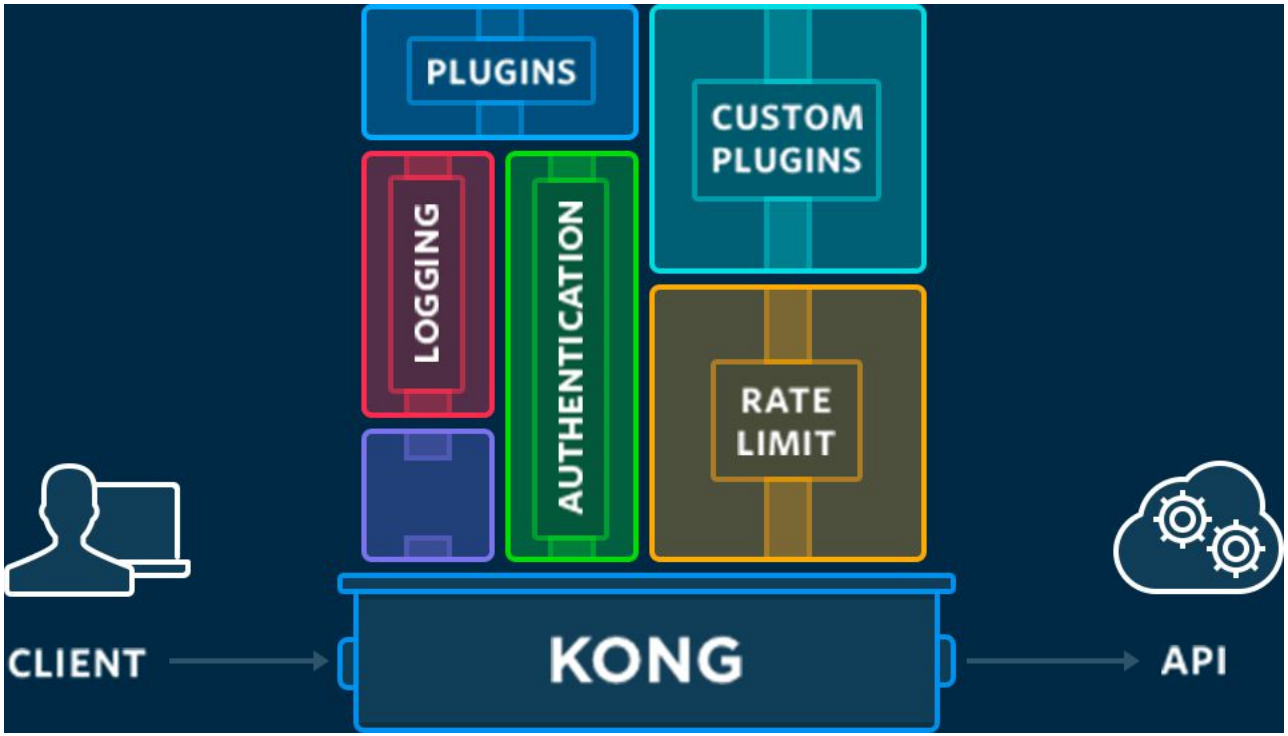
Workshop: Network Configure

- nginx.conf

```
upstream nodejs_web {
    server web1:3000;
    server web2:3000;
}

server {
        listen 8080 default_server;
        location /nodejs{
        proxy_pass http://nodejs_web;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        }
}

server {
        listen 8443 ssl http2;
        ssl_certificate         labdocker.com.crt;
        ssl_certificate_key     labdocker.com.key;
        location /nodejs{
        proxy_pass http://nodejs_web;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        }
}
```

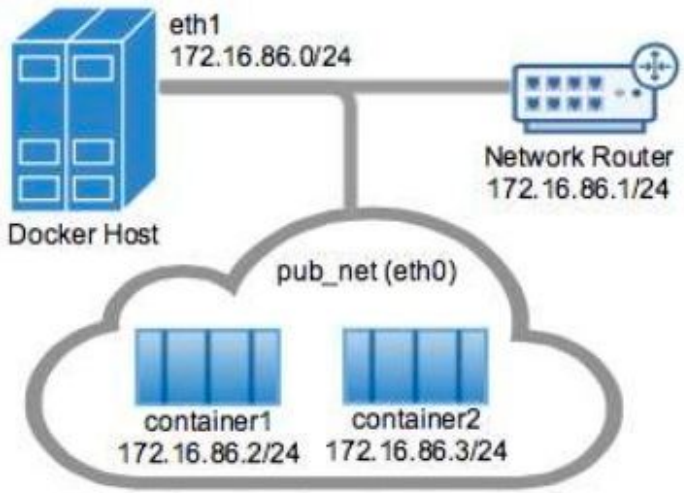
Workshop: Network Configure

- Bonus: Kong API Gateway the basic

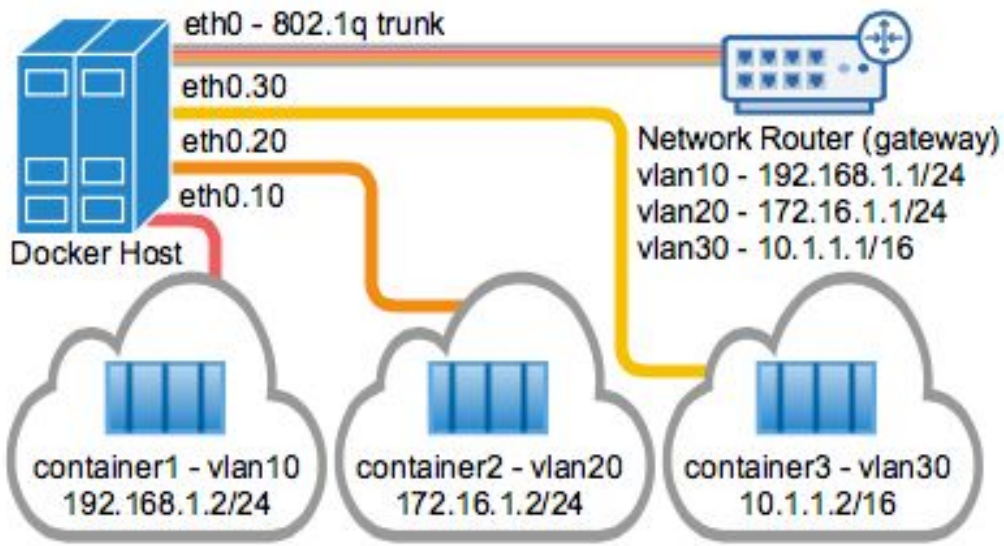


Network

- 802.1q Tagging (MACVLAN)




```
docker network create -d macvlan \
  –subnet=172.16.86.0/24 \
  –gateway=172.16.86.1  \
  -o parent=eth1 pub_net
```



Demo: MACVLAN

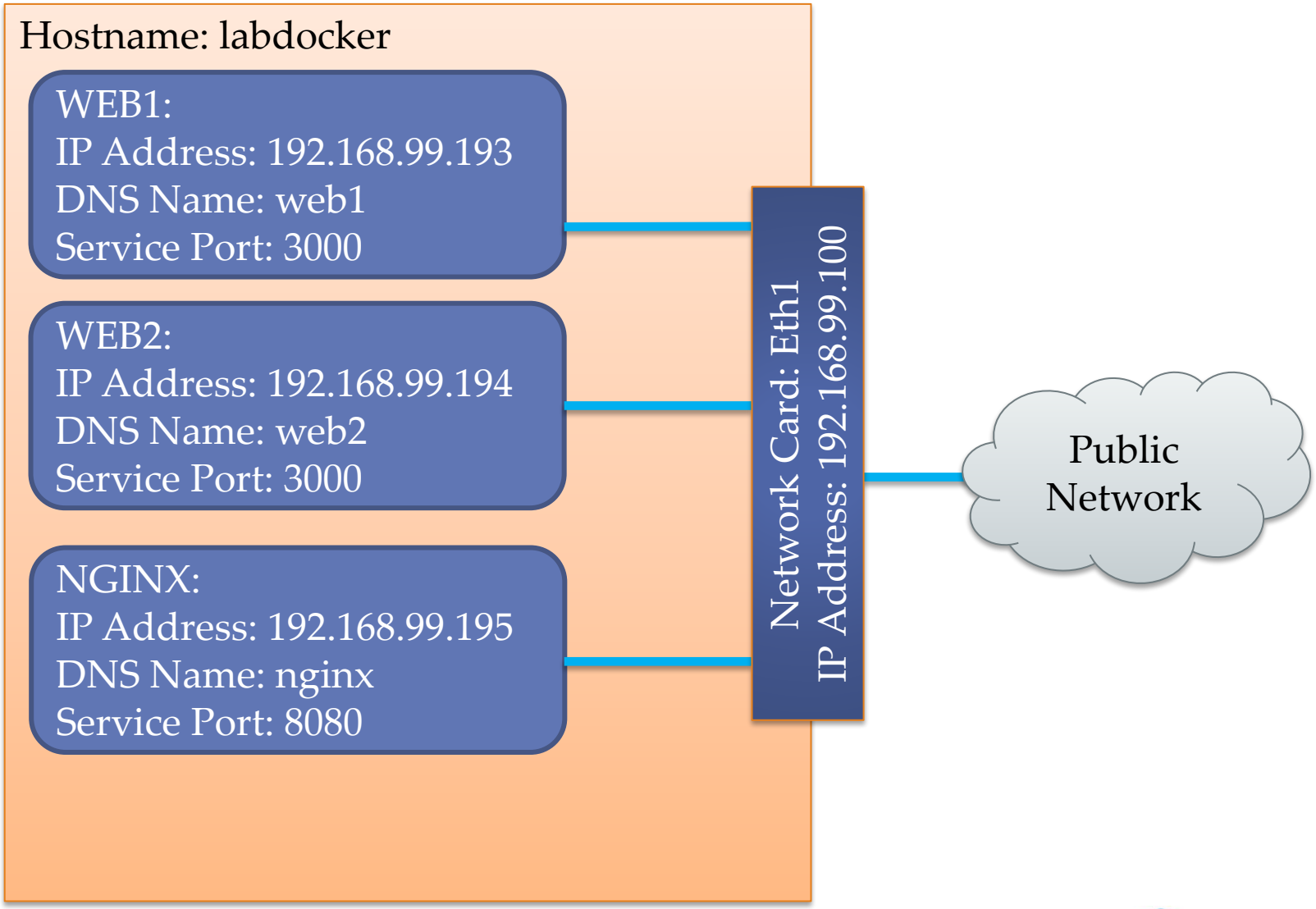
- Reverse Proxy Network (MACVLAN) (This need docker-machine)
- Purpose: Deploy nodejs webserver with solution proxy via Macvlan
- **Macvlan:**
  - Interface: eth1
  - IP Address: 192.168.99.0/24
  - Gateway: 192.168.99.1 ⇒ Your Machine
  - Labdocker: 192.168.99.100 ⇒ Default
  - Range Address: 192.168.99.192/28
    - 192.168.99.193 – 192.168.99.206

https://github.com/docker/libnetwork/blob/master/docs/macvlan.md

Medium jerkins vagrant Mesos Vue docker NGINX Taiwan Kubernetes PWA_Progressive_W... MYSQL_Cluster GeneralKB Nod

- **Note:** Linux Macvlan interface types are not able to ping or communicate with the default namespace IP address. For example, if you create a container and try to ping the Docker host's `eth0` it will **not** work. That traffic is explicitly filtered by the kernel to offer additional provider isolation and security. This is a common gotcha when a user first uses those Linux interface types since it is natural to ping local addresses when testing.

Demo: MACVLAN



Demo: MACVLAN

- Create virtual switch (MacVlan)
  - docker network create -d macvlan \
    --subnet=192.168.99.0/24 \
    --ip-range=192.168.99.192/28 \
    --gateway=192.168.99.1 \
    -o parent=eth1 macvlanlab

- (Option) Tag VLAN (802.1q) from Switch/Router to Docker Host as subinterface
  - docker network create -d macvlan \
    --subnet=192.168.99.0/24 \
    --ip-range=192.168.99.192/28 \
    --gateway=192.168.99.1 \
    -o parent=eth1.<vlan id> macvlanlab

Demo: MACVLAN

- สร้างเครื่อง Webserver (nodejs) และ Nginx
  - **Web1**
  - docker run -dt --name web1 --net macvlanlab \
    --net-alias web1 labdocker/alpineweb:web1 node hello.js

  - **Web2**
  - docker run -dt --name web2 --net macvlanlab \
    --net-alias web2 labdocker/alpineweb:web2 node hello.js

  - **NGINX**
  - docker run -dt --name nginx --net macvlanlab \
    labdocker/nginx:labnetworkhttp2

Demo: MACVLAN

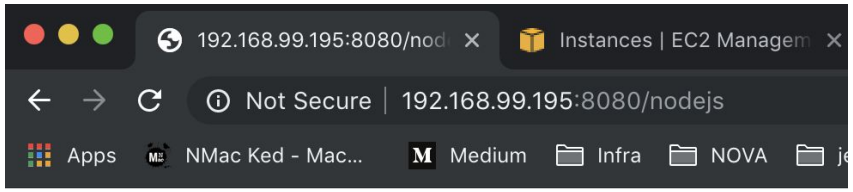
- Internal test:

```
...10-0-1-55: ~ — -bash ...   ...emultinodejs — -bash   ...ute.amazonaws.com ...   ...chine ssh labdocker
docker@labdocker:~$ docker container exec -it nginx curl http://web1:3000
Hello World Container Docker for Nodejs Number 1
docker@labdocker:~$ docker container exec -it nginx curl http://web1:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 03:39:07 GMT
Connection: keep-alive

docker@labdocker:~$ docker container exec -it nginx curl http://web2:3000
Hello World Container Docker for Nodejs Number 2
docker@labdocker:~$ docker container exec -it nginx curl http://web2:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 03:39:13 GMT
Connection: keep-alive

docker@labdocker:~$ docker container exec -it nginx ping web1
PING web1 (192.168.99.193): 56 data bytes
64 bytes from 192.168.99.193: seq=0 ttl=64 time=0.042 ms
^C
--- web1 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.042/0.042/0.042 ms
docker@labdocker:~$ docker container exec -it nginx ping web2
PING web2 (192.168.99.194): 56 data bytes
64 bytes from 192.168.99.194: seq=0 ttl=64 time=0.115 ms
64 bytes from 192.168.99.194: seq=1 ttl=64 time=0.069 ms
^C
--- web2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.069/0.092/0.115 ms
docker@labdocker:~$
```

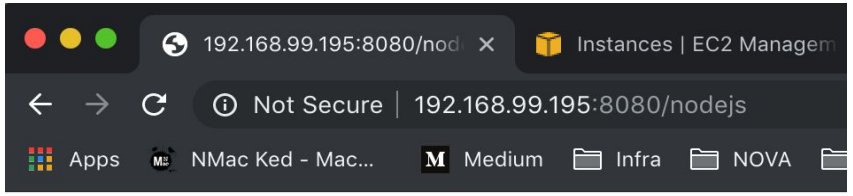
Demo: MACVLAN

- External Test:

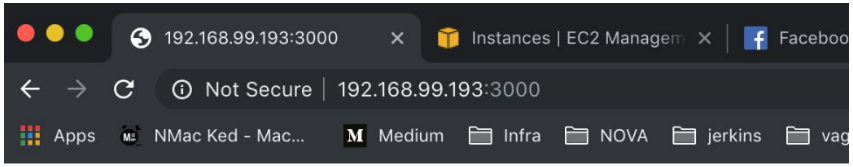
```
...10-0-1-55: ~ — -bash ...   ...emultinodejs — -bash   ...ute.amazonaws.com ...
praparns-MacBook-Pro:~ praparn$ ping 192.168.99.193
PING 192.168.99.193 (192.168.99.193): 56 data bytes
64 bytes from 192.168.99.193: icmp_seq=0 ttl=64 time=0.346 ms
64 bytes from 192.168.99.193: icmp_seq=1 ttl=64 time=0.508 ms
^C
--- 192.168.99.193 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.346/0.427/0.508/0.081 ms
praparns-MacBook-Pro:~ praparn$
praparns-MacBook-Pro:~ praparn$ ping 192.168.99.194
PING 192.168.99.194 (192.168.99.194): 56 data bytes
64 bytes from 192.168.99.194: icmp_seq=0 ttl=64 time=0.365 ms
64 bytes from 192.168.99.194: icmp_seq=1 ttl=64 time=0.470 ms
^C
--- 192.168.99.194 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.365/0.417/0.470/0.052 ms
praparns-MacBook-Pro:~ praparn$
praparns-MacBook-Pro:~ praparn$ ping 192.168.99.195
PING 192.168.99.195 (192.168.99.195): 56 data bytes
64 bytes from 192.168.99.195: icmp_seq=0 ttl=64 time=0.357 ms
64 bytes from 192.168.99.195: icmp_seq=1 ttl=64 time=0.331 ms
^C
--- 192.168.99.195 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.331/0.344/0.357/0.013 ms
praparns-MacBook-Pro:~ praparn$
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.195:8080/nodejs
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.195:8080/nodejs
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.193:3000
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.194:3000
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparn$ curl https://192.168.99.195:8443/nodejs -k
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparn$ curl https://192.168.99.195:8443/nodejs -k
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparn$
```



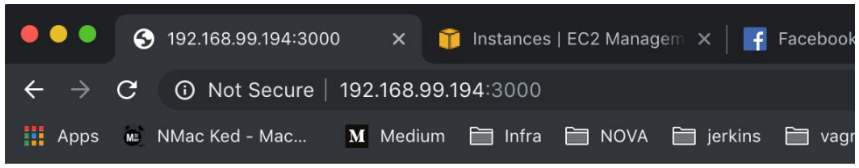
Hello World Container Docker for Nodejs Number 1



Hello World Container Docker for Nodejs Number 2



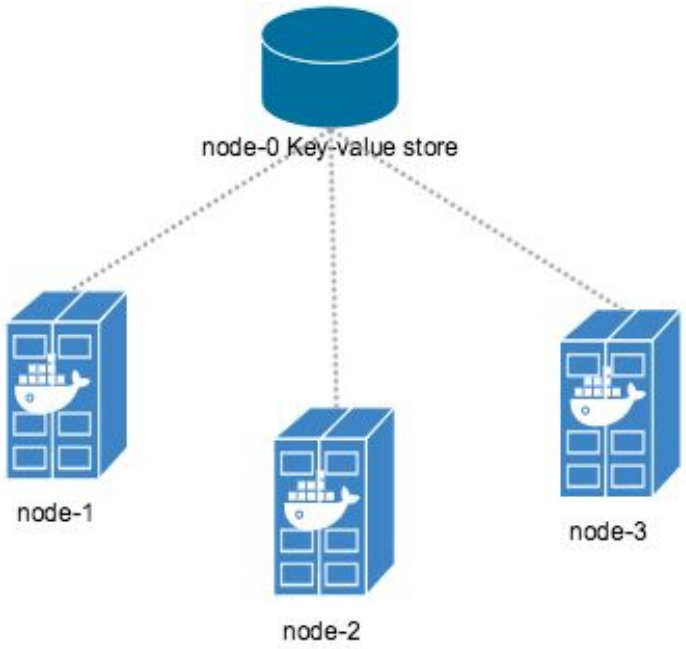
Hello World Container Docker for Nodejs Number 1



Hello World Container Docker for Nodejs Number 2

Network

- Network across host (Overlay Network)



Network

- 3rd Party Network Plugin

Plugin	Description
Contiv Networking	An open source network plugin to provide infrastructure and security policies for a multi-tenant micro services deployment, while providing an integration to physical network for non-container workload. Contiv Networking implements the remote driver and IPAM APIs available in Docker 1.9 onwards.
Kuryr Network Plugin	A network plugin is developed as part of the OpenStack Kuryr project and implements the Docker networking (libnetwork) remote driver API by utilizing Neutron, the OpenStack networking service. It includes an IPAM driver as well.
Weave Network Plugin	A network plugin that creates a virtual network that connects your Docker containers - across multiple hosts or clouds and enables automatic discovery of applications. Weave networks are resilient, partition tolerant, secure and work in partially connected networks, and other adverse environments - all configured with delightful simplicity.

Volume

Volume

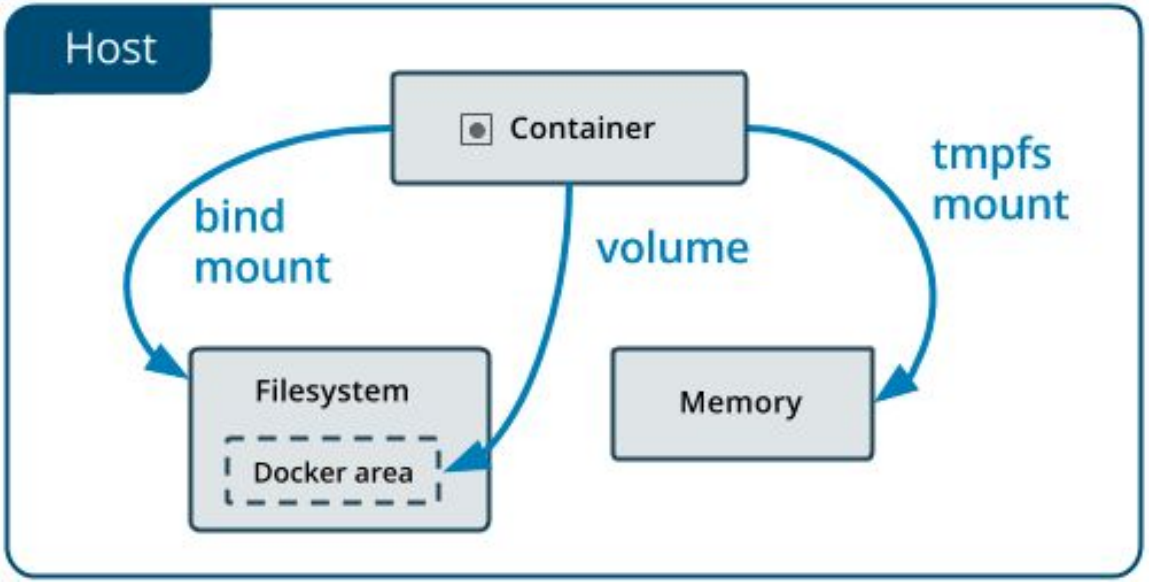
- (Default) Docker will housing all data inside container
  - /var/log
  - /sys/data
  - Etc.

- For use container in production environment. Separate data from container itself is very recommend proactive and can leverage in many case
  - Share data between container
  - Share data between container / host
  - Keep all application log on centralize
  - data migration
  - backup / restore

Volume

- Volume type:
  - Bind Mount (Map path with Host Directly)
  - Volumes Mount(Local, Special Driver)
  - Tmpfs Mount (Memory Only)

- Parameter (--mount), (-v (Obsolete))



Volume

- 1. Bind Mount: map volume between host directory and container directory

```
-v /volume-host:/volume-container:(ro/rw)
```

```
--mount type=bind,source=/volume-host
,target=/volume-container, (readonly)
```

Ex: docker container run –dt -v /etc/nginx.conf:/etc/nginx.conf:ro

Ex:docker container run -dt --mount type=bind,\
source=/etc/nginx.conf,target=/etc/nginx.conf,readonly

- Inspect configuration (inspect)

```
"HostConfig": {
        "Binds": [
           "/var/log:/var/log:ro"],
```

Volume

- 2. Volumes object (central) for share volume between host/container/container (local: /var/lib/docker/volumes,driver: 3rd Party)
  - Create Volume

```
docker volume create --driver <xxx> --opt <xxx> <name>
```

Ex: docker volume create --driver local datavol

```
Ex: docker volume create --driver local \
    --opt type=nfs --opt o=addr=192.168.99.100,rw \
    --opt device=:/nfs_share datavol
```

Volume

- Volumes (-v option is not allow for service level (Swarm/Compose))

Ex: docker volume create --driver local datavol

Ex: docker container run –dt --name web1 \
-v datavol:/data labdocker/alpine sh

Ex: docker container run –dt --name web2 \
-v datavol:/data labdocker/alpine sh

Ex: docker container run –dt --name web1 \
--mount source=datavol,target=/data labdocker/alpine sh

Ex: docker container run –dt --name web2 \
--mount source=datavol,target=/data labdocker/alpine sh

Volume

- Delete volume from container with command "docker volume rm"

```
docker volume rm <volume name>
```

- Third Party Volume Plugin

```
docker volume create --driver=<third party> <volume name>
```

```
Ex: docker volume create --driver=gce --name gce-disk\
 -o SizeGb=90
```

```
Ex: docker run –dt -v gce-disk:/store/database alpine sh
```

Volume

- NFS3:

```
docker service create -d \
  --name nfs-service \
  --mount
'type=volume,source=nfsvolume,target=/app,volume-driver=local,volume-opt=type=nfs,volume-opt=device=:/var/docker-nfs,volume-opt=o=addr=10.0.0.10' nginx:latest
```

- NFS4:

```
docker service create -d \
    --name nfs-service \
    --mount
'type=volume,source=nfsvolume,target=/app,volume-driver=local,volume-opt=type=nfs,volume-opt=device=:/var/docker-nfs,"volume-opt=o=addr=10.0.0.10,rw,nfsvers=4,async"' nginx:latest
```

Volume

- CIFS/SAMBA:

```
docker volume create \
--driver local \
--opt type=cifs \
--opt device=//uxxxxx.your-server.de/backup --opt
o=addr=uxxxxx.your-server.de,username=uxxxxxxx,password=*****,file_m
ode=0777,dir_mode=0777 --name cif-volume
```

Volume

lugin	Description
Azure File Storage plugin	Lets you mount Microsoft Azure File Storage shares to Docker containers as volumes using the SMB 3.0 protocol. Learn more.
BeeGFS Volume Plugin	An open source volume plugin to create persistent volumes in a BeeGFS parallel file system.
Blockbridge plugin	A volume plugin that provides access to an extensible set of container-based persistent storage options. It supports single and multi-host Docker environments with features that include tenant isolation, automated provisioning, encryption, secure deletion, snapshots and QoS.
Contiv Volume Plugin	An open source volume plugin that provides multi-tenant, persistent, distributed storage with intent based consumption. It has support for Ceph and NFS.
Convoy plugin	A volume plugin for a variety of storage back-ends including device mapper and NFS. It's a simple standalone executable written in Go and provides the framework to support vendor-specific extensions such as snapshots, backups and restore.
DigitalOcean Block Storage plugin	Integrates DigitalOcean's block storage solution into the Docker ecosystem by automatically attaching a given block storage volume to a DigitalOcean droplet and making the contents of the volume available to Docker containers running on that droplet.
DRBD plugin	A volume plugin that provides highly available storage replicated by DRBD. Data written to the docker volume is replicated in a cluster of DRBD nodes.
Flocker plugin	A volume plugin that provides multi-host portable volumes for Docker, enabling you to run databases and other stateful containers and move them around across a cluster of machines.
Fuxi Volume Plugin	A volume plugin that is developed as part of the OpenStack Kuryr project and implements the Docker volume plugin API by utilizing Cinder, the OpenStack block storage service.
gce-docker plugin	A volume plugin able to attach, format and mount Google Compute persistent-disks.
GlusterFS plugin	A volume plugin that provides multi-host volumes management for Docker using GlusterFS.
Horcrux Volume Plugin	A volume plugin that allows on-demand, version controlled access to your data. Horcrux is an open-source plugin, written in Go, and supports SCP, Minio and Amazon S3.
HPE 3Par Volume Plugin	A volume plugin that supports HPE 3Par and StoreVirtual iSCSI storage arrays.
IPFS Volume Plugin	An open source volume plugin that allows using an ipfs filesystem as a volume.
Keywhiz plugin	A plugin that provides credentials and secret management using Keywhiz as a central repository.
Local Persist Plugin	A volume plugin that extends the default local driver's functionality by allowing you specify a mountpoint anywhere on the host, which enables the files to *always persist*, even if the volume is removed via docker volume rm.
NetApp Plugin(nDVP)	A volume plugin that provides direct integration with the Docker ecosystem for the NetApp storage portfolio. The nDVP package supports the provisioning and management of storage resources from the storage platform to Docker hosts, with a robust framework for adding additional platforms in the future.
Netshare plugin	A volume plugin that provides volume management for NFS 3/4, AWS EFS and CIFS file systems.
Nimble Storage Volume Plugin	A volume plug-in that integrates with Nimble Storage Unified Flash Fabric arrays. The plug-in abstracts array volume capabilities to the Docker administrator to allow self-provisioning of secure multi-tenant volumes and clones.
OpenStorage Plugin	A cluster-aware volume plugin that provides volume management for file and block storage solutions. It implements a vendor neutral specification for implementing extensions such as CoS, encryption, and snapshots. It has example drivers based on FUSE, NFS, NBD and EBS to name a few.
Portworx Volume Plugin	A volume plugin that turns any server into a scale-out converged compute/storage node, providing container granular storage and highly available volumes across any node, using a shared-nothing storage backend that works with any docker scheduler.
Quobyte Volume Plugin	A volume plugin that connects Docker to Quobyte's data center file system, a general-purpose scalable and fault-tolerant storage platform.
REX-Ray plugin	A volume plugin which is written in Go and provides advanced storage functionality for many platforms including VirtualBox, EC2, Google Compute Engine, OpenStack, and EMC.
Virtuozzo Storage and Ploop plugin	A volume plugin with support for Virtuozzo Storage distributed cloud file system as well as ploop devices.
VMware vSphere Storage Plugin	Docker Volume Driver for vSphere enables customers to address persistent storage requirements for Docker containers in vSphere environmen

Volume

- Backup volume from container

Ex:

```
docker container run -it --rm --mount source=datavol,target=/data \
--mount type=bind,source=$(pwd) ,target=/backup \
tar cvf /backup/vol001.tar /data
```

- Restore volume to container

Ex:

```
docker container run  -it --rm --mount source=datavol,target=/data \
--mount type=bind,source=$(pwd) ,target=/backup \
bash -c "cd /data && tar xvf /backup/vol001.tar --strip 1"
```

Volume

- 3. tmpfs mount for keep data on memory for non-persistency space (tmpfs) that cannot share between container (not applicable on windows)

```
docker volume create --driver <xxx> --opt <xxx> <name>
```

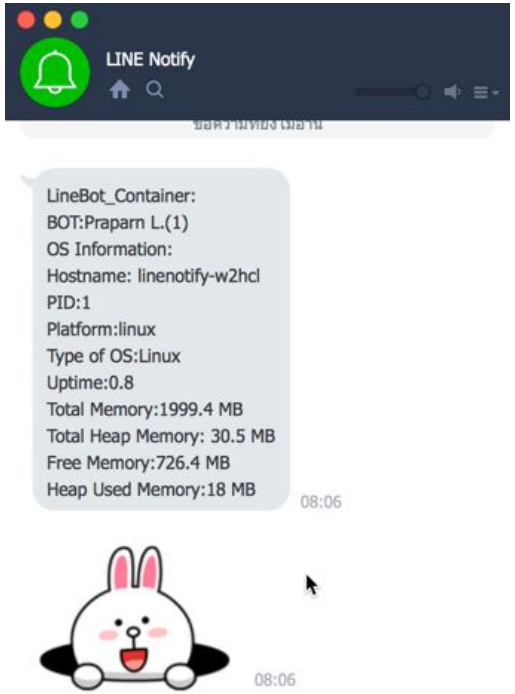
Ex: docker volume create --driver local --opt type=tmpfs \
--opt device=tmpfs --opt o=size=100m,uid=1000 tmptest

Ex: docker container run -dt --name tmptest –mount type=tmpfs,destination=/app nginx:latest

Ex: docker container run -dt --name tmptest --tmpfs /app \
nginx:latest

Workshop: Share Volume

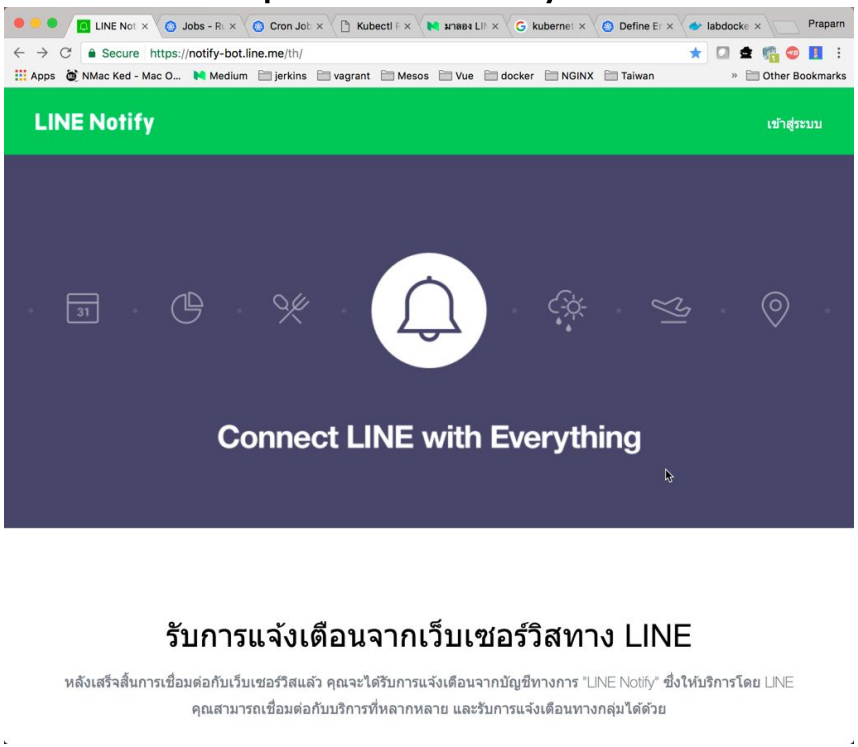
- <u>Part 1: Host Path debug program</u>
- Purpose: Map volume of source code for debug/run on container
- Example on WorkShop: ”Monitor and LINE notify

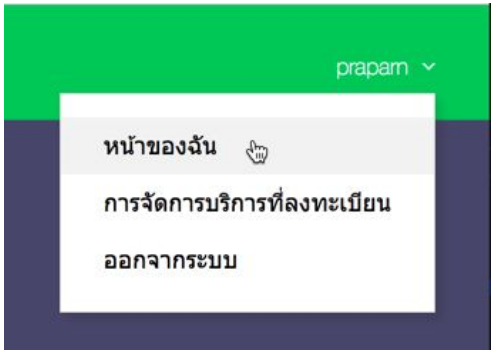


LINE

Workshop: Share Volume

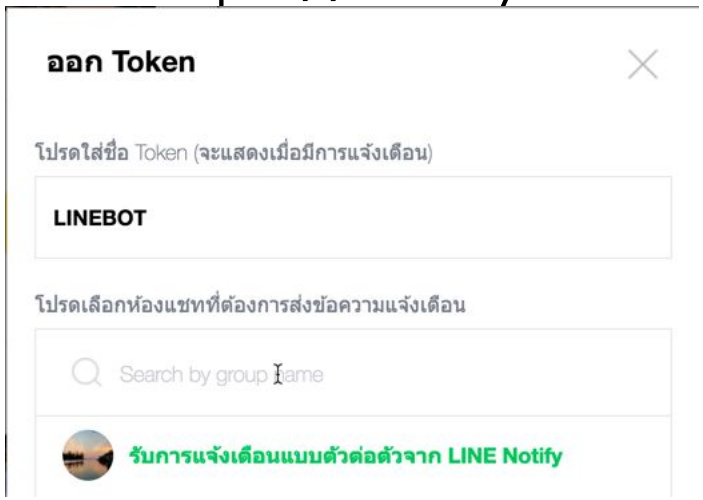
- <u>Generate LINE Token</u>
  - https://notify-bot.line.me

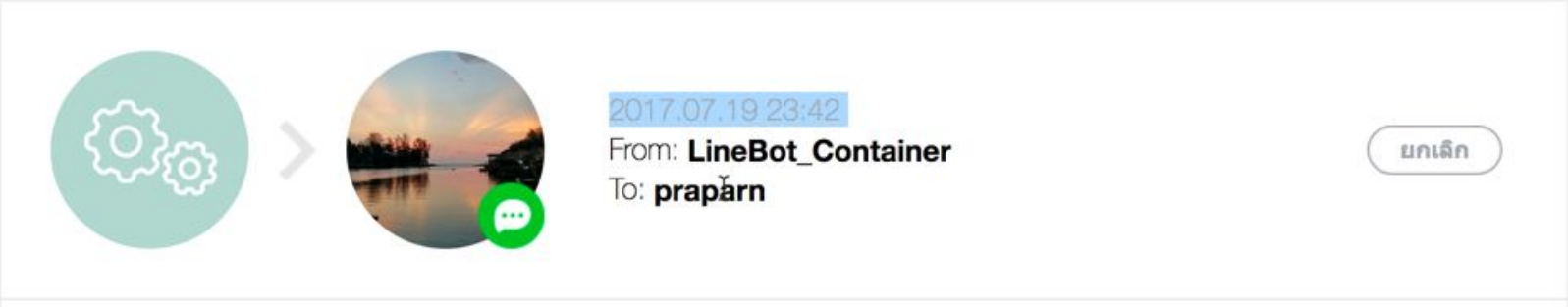




Workshop: Share Volume

- <u>Generate LINE Token</u>
  - https://notify-bot.line.me





Workshop: Share Volume

```
1   const Monitor = require('monitor');
2   const request = require('request');
3   const TITLE=process.env.TITLE;
4   const INTERVAL=process.env.INTERVAL;
5   const HEAP_HIGH =process.env.HEAP_HIGH;
6   const MEM_HIGH =process.env.MEM_HIGH;
7   const SH_OS =process.env.SH_OS;
8   const TOKEN = process.env.TOKEN;
9   const critical_StickerPkg = 2;
10  const critical_StickerId = 39;
11  const information_StickerPkg = 2;
12  const information_StickerId = 34;
13  var options = {
14    probeClass: 'Process',
15    initParams: {
16      pollInterval: INTERVAL
17    }
18  }
19  var processMonitor = new Monitor(options);
20  var stickerPkg=0; //stickerPackageId
21  var stickerId=0; //stickerId
22  processMonitor.on('change', () => {
23    var sendNotify="N" //final decision to send notify
24    var heapWarning="N" //flag for heap memory warning
25    var memoryWarning="N" //flag for memory warning
```

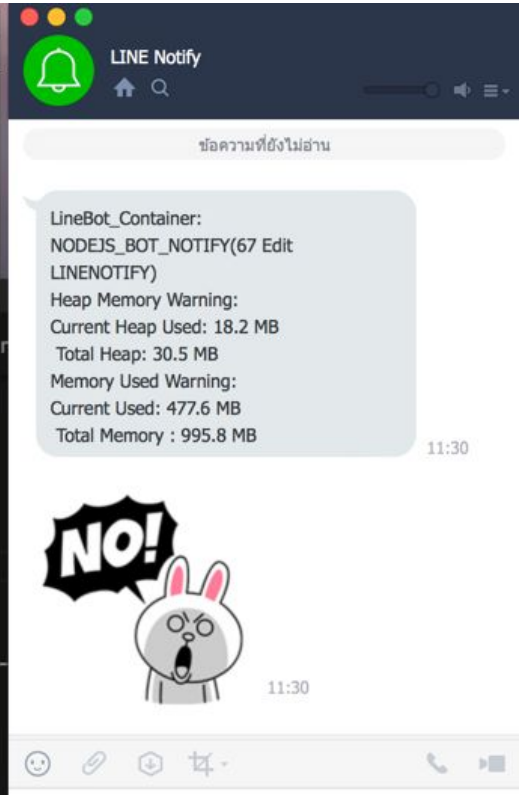
```
19  var processMonitor = new Monitor(options);
20  var stickerPkg=0; //stickerPackageId
21  var stickerId=0; //stickerId
22  processMonitor.on('change', () => {
23    var sendNotify="N" //final decision to send notify
24    var heapWarning="N" //flag for heap memory warning
25    var memoryWarning="N" //flag for memory warning
26    //collect OS information
27    var hostName = processMonitor.get('hostname');
28    var pID = processMonitor.get('pid');
29    var platForm = processMonitor.get('platform');
30    var upTime = processMonitor.get('uptime');
31    var type = processMonitor.get('type');
32    var totalMem = processMonitor.get('totalmem');
33    totalMem=Number(((totalMem/1024)/1024).toFixed(1)); //Convert to MB
34    var heapTotal = processMonitor.get('heapTotal');
35    heapTotal=Number(((heapTotal/1024)/1024).toFixed(1)); //Convert to MB
36    var titleMSG="\n"+TITLE+"("+pID+")"
37    var osMSG="";
38    var warnMSG="";
39    //collect OS information
40    //collect memory information
41    var heapUsed = processMonitor.get('heapUsed');
42    heapUsed=Number(((heapUsed/1024)/1024).toFixed(1)); //Convert to MB
43    if(HEAP_HIGH<((heapUsed/heapTotal)*100)){heapWarning="Y";}
44    var freeMem = processMonitor.get('freemem');
45    freeMem=Number(((freeMem/1024)/1024).toFixed(1));
46    var usedMem = totalMem-freeMem;
47    usedMem=Number(usedMem.toFixed(1)); //Convert to MB
48    if(MEM_HIGH<((usedMem/totalMem)*100)){memoryWarning="Y";}
49    //collect memory information
```

Workshop: Share Volume

praparnlueangphoonlap — Terminal MAC Pro — ssh ◂ docker-machine ssh labdocker — 102×25

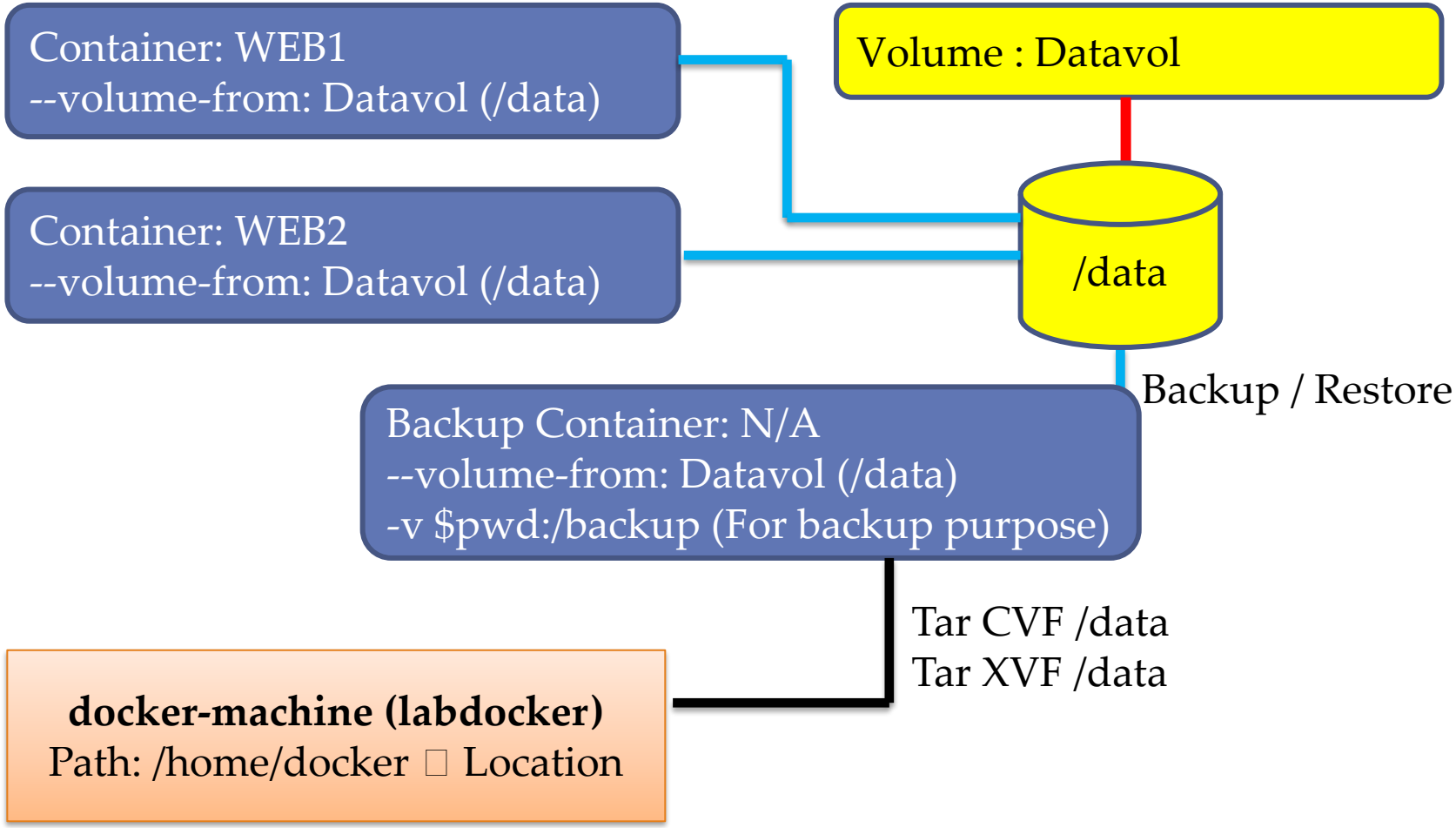
Terminal MAC Pro — ssh ◂ docker-machine ss... | ~ — Terminal MAC Pro — zsh | ~ — Terminal MAC Pro — -bash

```
mit":"50","x-ratelimit-remaining":"997","x-ratelimit-imageremaining":"50","x-ratelimit-reset":"1501996
936"},"request":{"uri":{"protocol":"https:","slashes":true,"auth":null,"host":"notify-api.line.me","po
rt":443,"hostname":"notify-api.line.me","hash":null,"search":null,"query":null,"pathname":"/api/notify
","path":"/api/notify","href":"https://notify-api.line.me/api/notify"},"method":"POST","headers":{"Con
tent-Type":"application/x-www-form-urlencoded","authorization":"Bearer EIeqyillzkzpaCo1ROrebZTcGbIyzDZ
HpOjcdd0t6CX","content-length":343}}}
"{\"status\":200,\"message\":\"ok\"}"
^C
/nodejs # export HEAP_HIGH=20
/nodejs # export MEM_HIGH=20
/nodejs # export SH_OS=N
/nodejs # node index.js
null
{"statusCode":200,"body":"{\"status\":200,\"message\":\"ok\"}","headers":{"server":"nginx","date":"Sun
, 06 Aug 2017 04:30:41 GMT","content-type":"application/json;charset=UTF-8","transfer-encoding":"chunk
ed","connection":"keep-alive","keep-alive":"timeout=3","x-ratelimit-limit":"1000","x-ratelimit-imageli
mit":"50","x-ratelimit-remaining":"996","x-ratelimit-imageremaining":"50","x-ratelimit-reset":"1501996
936"},"request":{"uri":{"protocol":"https:","slashes":true,"auth":null,"host":"notify-api.line.me","po
rt":443,"hostname":"notify-api.line.me","hash":null,"search":null,"query":null,"pathname":"/api/notify
","path":"/api/notify","href":"https://notify-api.line.me/api/notify"},"method":"POST","headers":{"Con
tent-Type":"application/x-www-form-urlencoded","authorization":"Bearer EIeqyillzkzpaCo1ROrebZTcGbIyzDZ
HpOjcdd0t6CX","content-length":290}}}
"{\"status\":200,\"message\":\"ok\"}"
^C
/nodejs #
```



Workshop: Share Volume

- Part 2: Container Volume



Recap Day 1

- ***Recap Container Concept**
- Container principle
- Image, Repository & Tag, container
- CPU, Memory and I/O
- Network
- Volume

Q&A

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

DAY 2

Outline Day 2

- ***Recap Container Concept**
- Inspect and Log
- Commit
- Dockerfile and Build
- Compose
- Docker Security
- Registry

Inspect/Log/Commit

Log

- Verity log from container with command:

```
docker container logs <options> <container name>
```

- options ในการดูล็อกไฟล์
  - -f, --follow
  - -t,--timestamps
  - --since= <unix timestamps>
  - --tail = “all” / number of line
  - --until (Ex: 20180619220001) or Relative (Ex: 42m = 42 minute before)
  - --details (Ex: Env variable, labels)

Log

- Server side. Big problem will come with logging !!!
- All container log will keep on "/var/lib/docker/containers/<container id>/<container-id>-json.log"and it take mega size on this path until full
- We have 2 option for manage this
- Global Configuration
  - Add daemon.json on path /etc/docker/daemon.json for global apply (need to restart docker daemon for take effect)

```
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m",
    "max-file": "10"
  },
  "storage-driver": "overlay2"
}
```

Log

Driver	Description
`none`	No logs are available for the container and `docker logs` does not return any output.
`local`	Logs are stored in a custom format designed for minimal overhead.
`json-file`	The logs are formatted as JSON. The default logging driver for Docker.
`syslog`	Writes logging messages to the `syslog` facility. The `syslog` daemon must be running on the host machine.
`journald`	Writes log messages to `journald`. The `journald` daemon must be running on the host machine.
`gelf`	Writes log messages to a Graylog Extended Log Format (GELF) endpoint such as Graylog or Logstash.
`fluentd`	Writes log messages to `fluentd` (forward input). The `fluentd` daemon must be running on the host machine.
`awslogs`	Writes log messages to Amazon CloudWatch Logs.
`splunk`	Writes log messages to `splunk` using the HTTP Event Collector.
`etwlogs`	Writes log messages as Event Tracing for Windows (ETW) events. Only available on Windows platforms.
`gcplogs`	Writes log messages to Google Cloud Platform (GCP) Logging.
`logentries`	Writes log messages to Rapid7 Logentries.

Ref https://docs.docker.com/config/containers/logging/configure/

Log

- Container Configuration
  - Specific on run command of docker container
    - --log-driver json-file (default)
    - --log-opt max-size=10m
    - --log-opt max-file=10

```
ubuntu@ip-10-0-1-55:~$ docker container run -dt --name nginx -p 80:80 \
> --log-driver json-file \
> --log-opt max-size=10m \
> --log-opt max-file=10 \
> labdocker/nginx:badversion
07af88d6368bdfd8a74608882963a50052e5c887fc32da812369d78068643e51
ubuntu@ip-10-0-1-55:~$
```

Inspect

- Check container / network / volume etc

```
docker inspect <container >
```

```
docker network inspect
```

```
docker volume inspect
```

Workshop: Log & Inspect

- Create container nginx and map port 80

```
docker container run -dt –name nginx \
–p 80:8080 labdocker/nginx: badversion
```

- Inspect container configuration

```
docker container inspect nginx
```

- Check container log

```
docker container logs nginx
```

Commit

- For finish running container and package it to image. We need to use the "commit"

```
docker container commit  <options>  <container> <image name:tag>
```

- options
  - -a, --author=" "
  - -c, --change = [ ]
  - -m, --message = " "
  - -p, --pause=true

DockerFile and Build

Dockerfile

- docker file is standard instruction file for build docker image
  - FROM
  - MAINTAINER
  - RUN (shell), [“exec”, “parameter1”,”parameter2”,”etc”]
  - CMD (shell),[“exec”,”parameter1”,”parameter2”,”etc”]
  - EXPOSE
  - ARGS
  - ENV
  - COPY /ADD (source) (Destination), [(source) (destination)]
  - ENTRYPOINT
  - WORKDIR
  - HEALTHCHECK
  - Etc(https://docs.docker.com/engine/reference/builder)
- Build docker by command:

```
docker build <option> <path of dockerfile>
```

Dockerfile

- .dockerignore file exclude some file/folder that not sent to build in image

```
# comment
*/temp*
*/*/temp*
temp?
*.md
!README*.md
```

Dockerfile

- Option for docker build
  - --add-host Add customer /etc/hosts
  - --build-arg Set build-time variables
  - --cache-from Image to consider
  - --cgroup-parent Specific parent control group
  - --compress Compress context with gzip
  - --cpu-shares CPU shares
  - --cpuset-cpus CPU no. for use
  - --memory, -m Memory limit
  - --disable-content -trust image verification
  - --file, -f Specify docker file
  - --rm Remove intermediate image after build finished
  - --force-rm Force remove intermediate image
  - --network Set network in build operate (RUN)
  - --no-cache Never use cache
  - --pull Always attempt pull new version image
  - --tag, -t Tag image name

Ref: https://docs.docker.com/engine/reference/commandline/build/

Dockerfile

- **Best practice for dockerfile**
- simple is the best
- 1 container / 1 service
- source image from official owner
- Create dockerfile with minimum footprint
  - Create path only need for application
  - Place only file on application
  - install component only need container
- Conclude all install command with single line
- Remove temporary file during build process
- Rearrange dockerfile to easy to review
  - apt-update && apt-install –y \
  - curl \
  - automake \
- EXPOSE port for application need

Ref: https://docs.docker.com/develop/develop-images/dockerfile_best-practices/

Dockerfile

```
RUN apt-get update && apt-get install -y \
    aufs-tools \
    automake \
    build-essential \
    curl \
    dpkg-sig \
    libcap-dev \
    libsqlite3-dev \
    mercurial \
    reprepro \
    ruby1.9.1 \
    ruby1.9.1-dev \
    s3cmd=1.1.* \
 && rm -rf /var/lib/apt/lists/*
```

Dockerfile

```
FROM php:5.6-apache
MAINTAINER Volker Wiegand <volker.wiegand@cvw.de>

RUN a2enmod rewrite

RUN apt-get update && apt-get install -y \
        libpng12-dev \
        libjpeg-dev \
        libpq-dev \
        libxml2-dev \
        vim-tiny \
        && docker-php-ext-configure gd --with-png-dir=/usr --with-jpeg
        && docker-php-ext-install gd mbstring mysql mysqli pgsql soap
        && rm -rf /var/lib/apt/lists/* /var/www/html/index.html

ENV MANTIS_VER 1.3.2
ENV MANTIS_MD5 f30acb6d41ba757b7c09f922a0f68b06
ENV MANTIS_URL https://sourceforge.net/projects/mantisbt/files/mantis-
ENV MANTIS_FILE mantisbt.tar.gz

RUN mkdir -p /var/lib/mantisbt && cd /var/lib/mantisbt \
        && curl -fSL ${MANTIS_URL} -o ${MANTIS_FILE} \
        && echo "${MANTIS_MD5}  ${MANTIS_FILE}" | md5sum -c \
        && tar -xz --strip-components=1 -f ${MANTIS_FILE} \
        && rm ${MANTIS_FILE} \
        && chown -R www-data:www-data .
```

Dockerfile

- Define WORKDIR for setup start path (before ENTRYPOINT, RUN, CMD, COPY etc)
- ADD/COPY operation
  - COPY basic transfer simple file
  - ADD remote file URL, self extract (.tar) (Not recommend)
- ENV PATH specify only necessary for application
- Consideration cache for image layer and bypass cache with option “--no-cache=true”
- Define HealthCheck for verity application

Docker Image Version

- Docker start to introduce standard of image manifest and schema on version 1.13.0 with version1 (deprecate)
- So standard image version is base on docker version that install on machine to build image
- In case that we run the deprecate version of docker image (schema v1). Docker will show warning during start container
- Converting image manifest is possible just “docker image pull” and “docker image push” from docker machine with up-to-date version. Docker will convert image format automatically
- Current docker image have 3 release of manifest
  - Manifest v1, Schema v1 (Deprecate)
  - Manifest v2, Schema v1 (Many vulnerability)
  - Manifest v2, Schema v2 (Recommend)

Docker Image Version

- Main feature of "Manifest v2, Scheme v2
  - Support multi-architecture (fat manifest) image for contain single image
  - Keep image configuration to hash as id and embed on image
- Verify manifest and schema version by command:

```
docker manifest inspect <image version>
```

  - application/vnd.docker.distribution.manifest.v1+json
  - application/vnd.docker.distribution.manifest.v2+json
  - application/vnd.docker.distribution.manifest.list.v2+json
  - application/vnd.docker.container.image.v1+json
  - application/vnd.docker.image.rootfs.diff.tar.gzip
  - application/vnd.docker.image.rootfs.foreign.diff.tar.gzip
  - application/vnd.docker.plugin.v1+json

Docker Image Version

- Verify manifest and schema version by command:

```
ubuntu@ip-10-0-1-65:~$ docker manifest inspect labdocker/alpine:latest
{
        "schemaVersion": 2,
        "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
        "config": {
                "mediaType": "application/vnd.docker.container.image.v1+json",
                "size": 1472,
                "digest": "sha256:7731472c3f2a25edbb9c085c78f42ec71259f2b83485aa60648276d408865839"
        },
        "layers": [
                {
                        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
                        "size": 2810825,
                        "digest": "sha256:596ba82af5aaa3e2fd9d6f955b8b94f0744a2b60710e3c243ba3e4a467f051d1"
                }
        ]
}
ubuntu@ip-10-0-1-65:~$
```

How can check dockerfile ?

Praparn

Secure https://www.fromlatest.io/#/

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

FROM:latest

Analyze | About | Contribute

```
1  FROM labdocker/alpine:latest
2  MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
3  LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
4  ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
5  RUN apk update && \
6      apk add nginx
7  COPY nginx.conf /etc/nginx/nginx.conf
8  WORKDIR /usr/sbin
9  ENTRYPOINT ["nginx","-c","/etc/nginx/nginx.conf"]
10 EXPOSE 8080
```

Line 5: Consider `–no-cache or –update with rm -rf /var/cache/apk/*` (Optimization)

Line 1: Base Image Latest Tag (Clarity)

2 issues found

Made by REPLICATED

Instruction_Kubernet....txt | Instruction_Kubernet....txt | Show All

kubernetes by Google

Workshop: DockerFile

- Part1: SCRATCH

docker-library / hello-world

Watch 29 | Star 214 | Fork 260

Code | Issues 0 | Pull requests 1 | Projects 0 | Wiki | Insights

No description, website, or topics provided.

68 commits | 1 branch | 0 releases | 9 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

yosifkit Merge pull request #54 from infosiftr/1809 — Latest commit b9918f5 on Jan 1, 2019

amd64	Add Windows 1809 support	2 months ago
arm32v5	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
arm32v7	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
arm64v8	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
greetings	Add initial multiarchitecture support via Debian cross-compiled binaries	2 years ago
i386	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
ppc64le	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
s390x	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
.dockerignore	Add Nano Server 1803 variants	8 months ago
.travis.yml	Use new microsoft/nanoserver tags (sac2016)	a year ago
Dockerfile-linux.template	Add Nano Server 1803 variants	8 months ago
Dockerfile-windows.template	Add Windows 1809 support	2 months ago
Dockerfile.build	Add i386!	2 years ago
LICENSE	Remove ending year from LICENSE	7 months ago
Makefile	Add Windows 1809 support	2 months ago
README.md	Update generated README	11 months ago
generate-stackbrew-library.sh	Add Windows 1809 support	2 months ago
hello.c	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
update.sh	Add Windows 1809 support	2 months ago

README.md

https://github.com/docker-library/hello-world

Maintained by: the Docker Community

This is the Git repo of the Docker "Official Image" for hola-mundo (not to be confused with any official hola-mundo image provided by hola-mundo upstream). See the Docker Hub page for the full readme on how to use this Docker image and for information regarding contributing and issues.

The full description from Docker Hub is generated over in docker-library/docs, specifically in docker-library/docs/hola-mundo.

```
instruction.txt  nginx  nodejs  python_restfulset  scratch
ubuntu@ip-10-0-1-104:~$ docker image build -t labdocker/hello-world:1.0 ~/docker-workshop-042019/Workshop-2-1-DockerFile/scratch/
Sending build context to Docker daemon   16.9kB
Step 1/3 : FROM scratch
 --->
Step 2/3 : COPY hello /
 ---> Using cache
 ---> 937200b29047
Step 3/3 : CMD ["/hello"]
 ---> Using cache
 ---> e759df7ef1b1
Successfully built e759df7ef1b1
Successfully tagged labdocker/hello-world:1.0
ubuntu@ip-10-0-1-104:~$ docker container run labdocker/hello-world:1.0

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

ubuntu@ip-10-0-1-104:~$
```

Workshop: DockerFile

- Part2: NGINX/NODEJS

Container Name: NODEJS
IP Address: X.X.X.X (Port: 3000:3000)

Container Name: NGINX
IP Address: X.X.X.X (Port: 8080:80)

AWS's Machine
Path: ~/dockerworkshop/Workshop-2-1-DockerFile

/nodejs

dockerfile

/nginx

dockerfile

Workshop: DockerFile

- Verify dockerfile as example below

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v14.15.4 NPM_VERSION=v14.15.4
RUN apk update && \
    apk add nodejs nodejs-npm curl --no-cache && \
    rm -rf /var/cache/apk/*
RUN mkdir /nodejs
COPY hello.js /nodejs/
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:3000/ || exit 1
```

docker image build -t labdocker/node:1.0 ~/docker-workshop-072019/Workshop-2-1-DockerFile/nodejs

Workshop: DockerFile

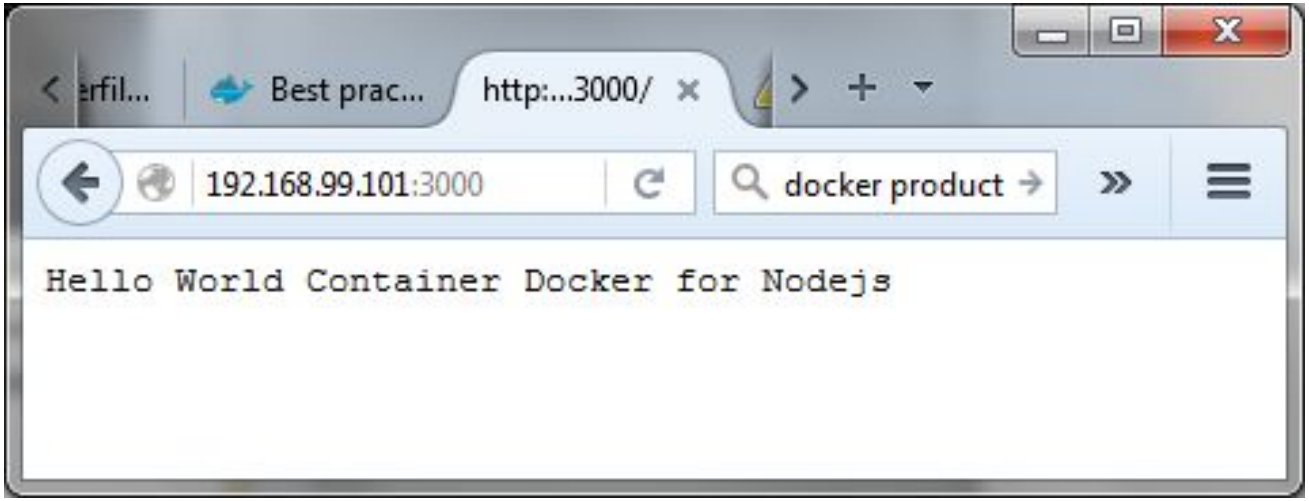
- Output

```
Sending build context to Docker daemon 3.072 kB
Step 1 : FROM labdocker/alpine:latest
 ---> 14f89d0e6257
Step 2 : MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
 ---> Using cache
 ---> 10f80fb4997d
Step 3 : LABEL Description "NodeJS/NGINX Build Container" Version "1.0"
 ---> Using cache
 ---> 59bc5a927e19
Step 4 : ENV NODE_VERSION v4.3.0 NPM_VERSION 2.14.12
 ---> Using cache
 ---> ce23d23959b7
Step 5 : RUN apk update &&     apk add nodejs
 ---> Running in b12c752ef6a0
fetch http://dl-4.alpinelinux.org/alpine/v3.3/main/x86_64/APKINDEX.tar.gz
fetch http://dl-4.alpinelinux.org/alpine/v3.3/community/x86_64/APKINDEX.tar.gz
v3.3.1-99-gc7d905f [http://dl-4.alpinelinux.org/alpine/v3.3/main]
v3.3.1-59-g48b0368 [http://dl-4.alpinelinux.org/alpine/v3.3/community]
OK: 5857 distinct packages available
(1/4) Installing libgcc (5.3.0-r0)
(2/4) Installing libstdc++ (5.3.0-r0)
(3/4) Installing libuv (1.7.5-r0)
(4/4) Installing nodejs (4.3.0-r0)
Executing busybox-1.24.1-r7.trigger
OK: 29 MiB in 15 packages
 ---> 8f2bf208ab86
Removing intermediate container b12c752ef6a0
Step 6 : RUN mkdir /nodejs
 ---> Running in 2ee51d1af642
 ---> d3edf2c2f511
Removing intermediate container 2ee51d1af642
Step 7 : COPY hello.js /nodejs/
 ---> f703ccbc7dd4
Removing intermediate container 952935d69cce
Step 8 : CMD nginx -c /etc/nginx/nginx.conf
 ---> Running in 67940385f5ac
 ---> edfba609e20a
```

Workshop: DockerFile

- Test run

```
docker container run -dt --name NODEJS \
-p 3000:3000 labdocker/node:1.0
```



Workshop: DockerFile

- Verify dockerfile as example below

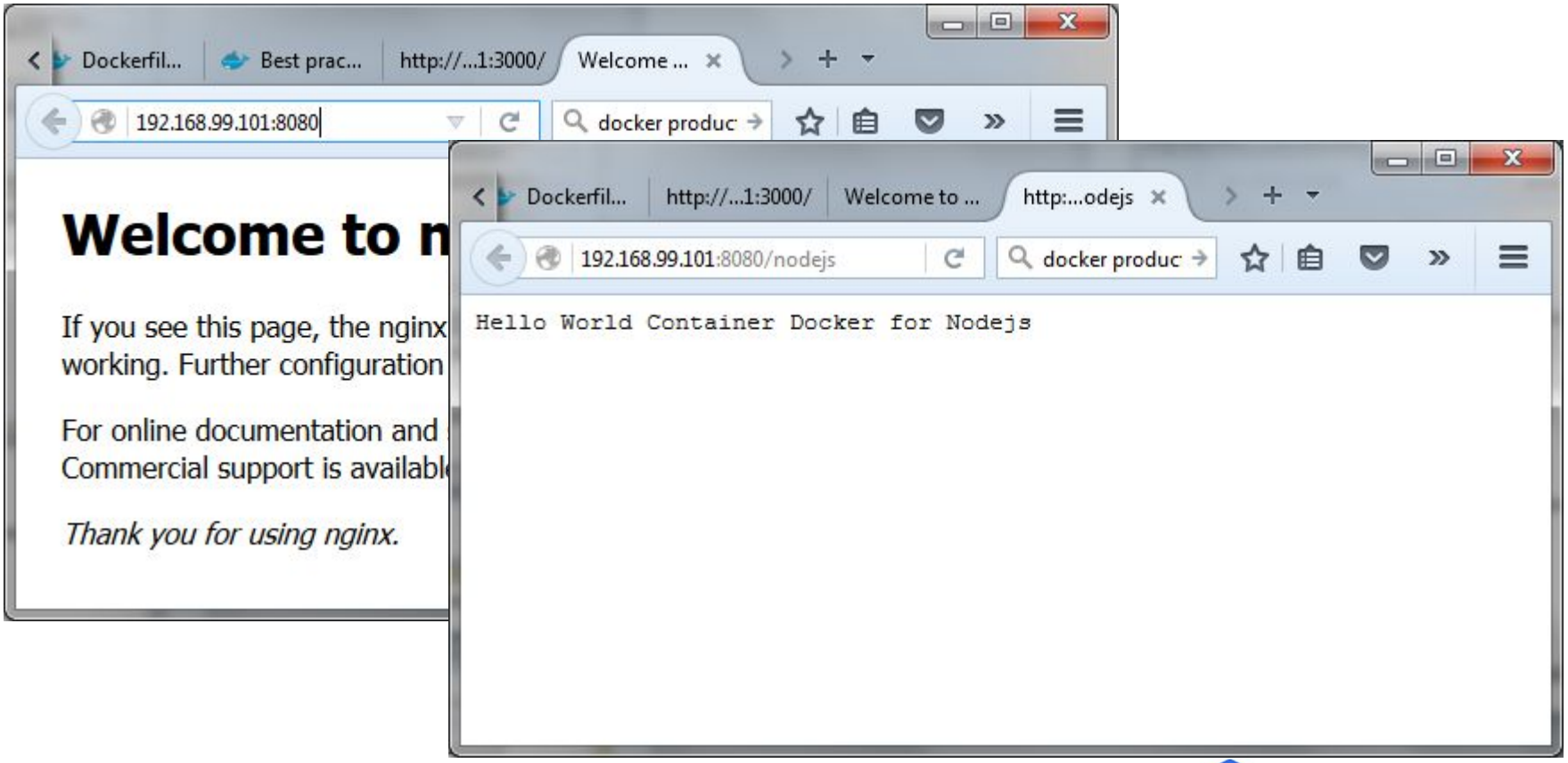
```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v14.15.4 NPM_VERSION=v14.15.4
RUN mkdir -p /run/nginx
RUN apk update && \
    apk add nginx curl --no-cache && \
    rm /etc/nginx/conf.d/default.conf && \
    rm -rf /var/cache/apk/*
COPY nginx.conf /etc/nginx/nginx.conf
WORKDIR /usr/sbin
ENTRYPOINT ["nginx","-c","/etc/nginx/nginx.conf"]
EXPOSE 8080
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:8080/ || exit 1
```

docker build -t labdocker/web:1.0
~/docker-workshop-072019/Workshop-2-1-DockerFile/nginx

Workshop: DockerFile

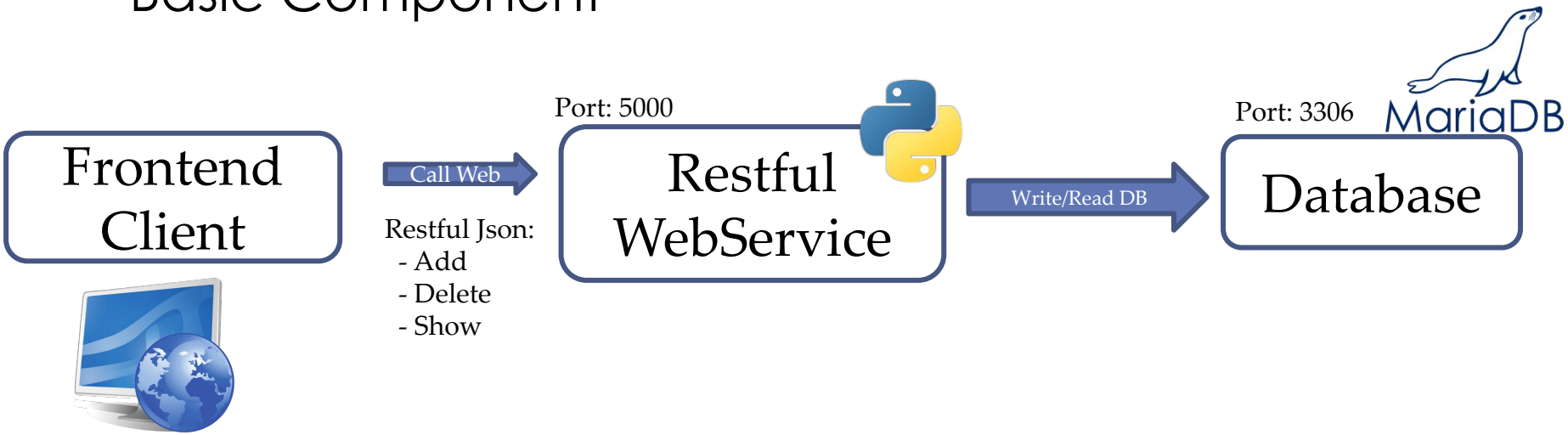
- Test run

```
docker container run -dt --name NGINX -p 8080:8080 -p
8443:8443labdocker/web:1.0
```



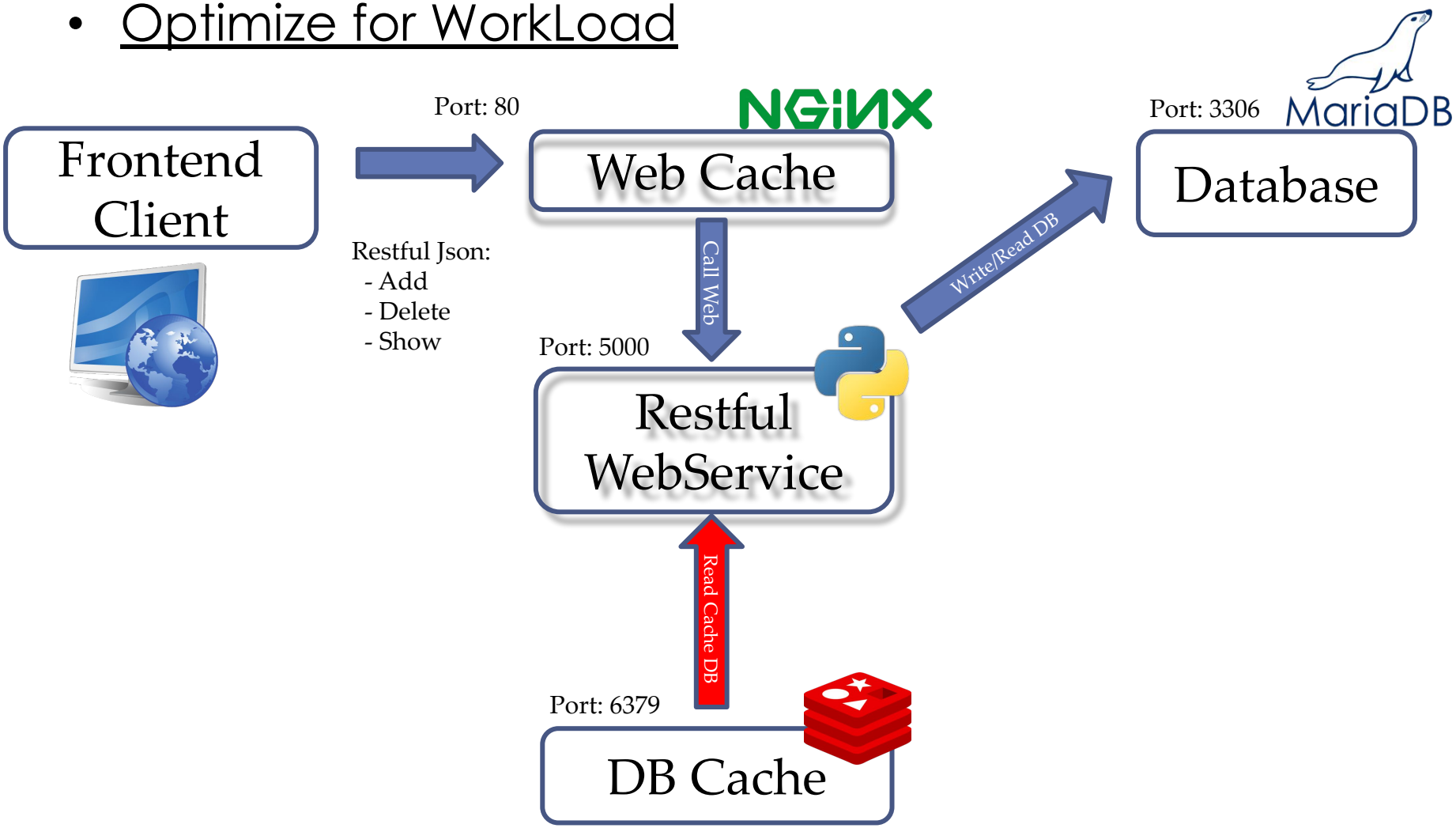
Workshop: DockerFile

- Part3: Python RESTFUL
- Basic Component



Workshop: DockerFile

- Optimize for WorkLoad



kubernetes by Google

Workshop: DockerFile

- 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')
```

Workshop: DockerFile

- 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')
```

Workshop: DockerFile

- 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 ###########"
```

Workshop: DockerFile

- Web Cache (NGINX)

```
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;
        }
        }
```

Workshop: DockerFile

- Database / 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
```

Maria Database

Redis Key-Value Database

Workshop: DockerFile

- Dockerfile: WebService

```
FROM labdocker/alpinepython:2.7-onbuild
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
RUN apk update && \
    apk add curl --no-cache && rm -rf /var/cache/apk/*
EXPOSE 5000
CMD [ "python", "main.py" ]
HEALTHCHECK --interval=30s --timeout=5s --start-period=30s --retries=3 \
    CMD curl -f http://localhost:5000/ || exit 1
```

docker build -t labdocker/webservice:1.0 \
-f dockerfile_python Path:
~/dockerworkshop/Workshop-2-1-DockerFile/python_restfulset
/

Workshop: DockerFile

- Dockerfile: Webcache

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
RUN apk update && \
    apk add nginx nginx curl --no-cache && rm -rf /var/cache/apk/*
COPY nginx.conf /etc/nginx/nginx.conf
WORKDIR /usr/sbin
ENTRYPOINT ["nginx","-c","/etc/nginx/nginx.conf"]
EXPOSE 8080
HEALTHCHECK --interval=30s --timeout=5s --start-period=30s --retries=3 \
    CMD curl -f http://localhost:8080/ || exit 1
```

```
docker build -t labdocker/webcache:1.0 \
-f dockerfile_nginx
~/dockerworkshop/Workshop-2-1-DockerFile/python_restfulset/
```

Workshop: DockerFile

- Test run container

```
ubuntu@ip-10-0-1-90:~/docker-workshop-092021/Workshop-2-1-DockerFile/python_restfulset$ docker container ls
CONTAINER ID   IMAGE                    COMMAND                  CREATED          STATUS                  PORTS                                          NAMES
ab037be198e2   labdocker/webcache:1.0   "nginx -c /etc/nginx…"   43 seconds ago   Up 42 seconds (healthy) 0.0.0.0:80->8080/tcp, :::80->8080/tcp          webcache
e7b40d9b3683   labdocker/webservice:1.0 "python main.py"         13 minutes ago   Up 13 minutes (healthy) 0.0.0.0:5000->5000/tcp, :::5000->5000/tcp      webservice
00b582c4da1d   labdocker/redis:latest   "docker-entrypoint.s…"   14 minutes ago   Up 14 minutes           6379/tcp                                       cachedb
82b1e278d02f   labdocker/mariadb:10.5.12 "docker-entrypoint.s…"  14 minutes ago   Up 14 minutes           3306/tcp                                       maindb
ubuntu@ip-10-0-1-90:~/docker-workshop-092021/Workshop-2-1-DockerFile/python_restfulset$
```

```
.ssh — ubuntu@ip-10-0-1-90: ~/docker-workshop-092021 — ssh -i docker_lab.pem ubuntu@ec2-54-254-231-32.ap-southeast-1.compute.amazonaws.com — 187×31
...092021 — Terminal MAC Pro — -bash   ...cesetup — Terminal MAC Pro — -bash   ...eacher — Terminal MAC Pro — -bash   ...utheast-1.compute.amazonaws.com   ...utheast-1.compute.amazonaws.com
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl ifconfig.co
54.254.231.32
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ export Server_IP=54.254.231.32
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ export Server_Port=80
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Fri Sep  3 18:33:20 2021
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$  curl http://$Server_IP:$Server_Port/init
###########   Database Create New Account Table Done  ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$
```

Not Secure | 54.254.231.32

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

Welcome Page from Container Python Lab

Checkpoint Date/Time: Fri Sep 3 18:33:56 2021

Workshop: DockerFile

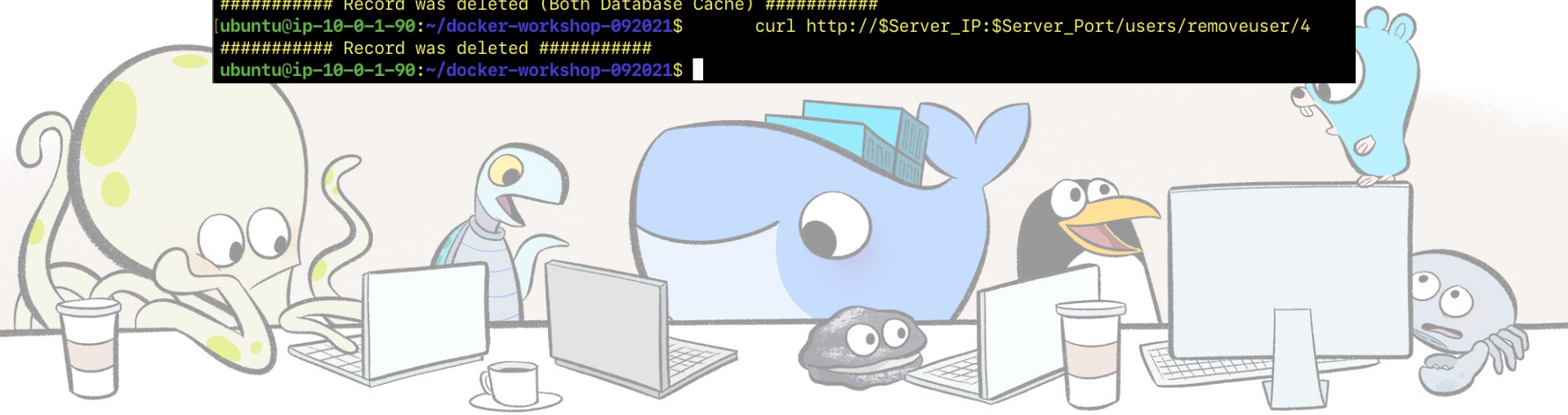
- Init / Insert / Delete Data

```
########### Record was added ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "8", "user":"Pansa Bunsong", "descripe":"Secuirty Guard"}' http://$Server_IP:$Server_Port/users/insertuser
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Fri, 03 Sep 2021 18:34:55 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive

########### Record was added ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "9", "user":"Wiphanee Wongsaisawan", "descripe":"Administrator"}' http://$Server_IP:$Server_Port/users/insertuser
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Fri, 03 Sep 2021 18:34:57 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive

########### Record was added ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/removeuser/1
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/2
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/3
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/4
########### Record was deleted ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$
```

Workshop: DockerFile

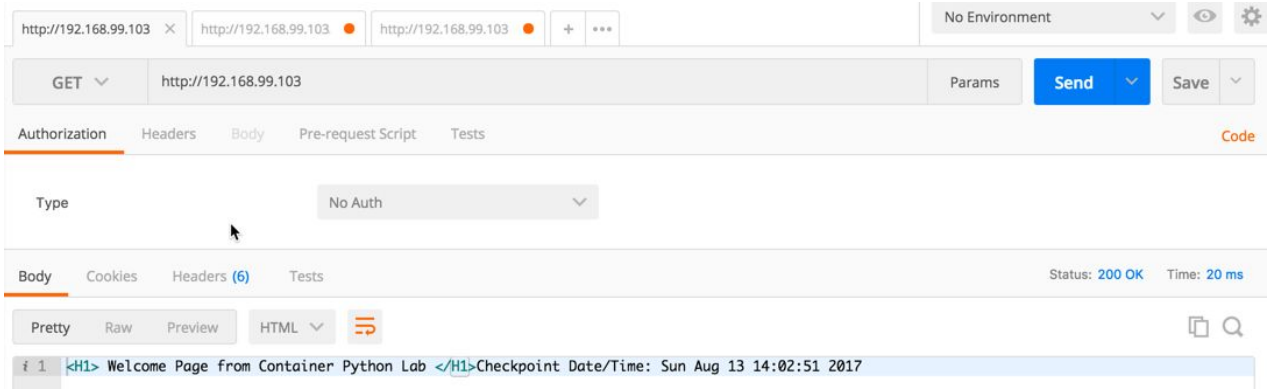
- Get Data (Direct/Cache) and Delete Data

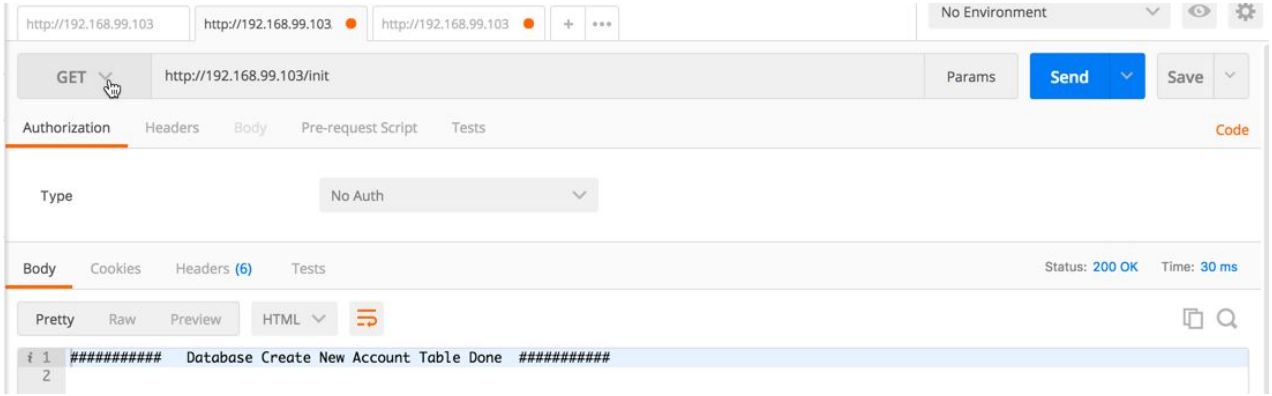
```
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/removeuser/1
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/2
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/3
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/4
########### Record was deleted ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$
```



Workshop: DockerFile

- Case for POSTMAN





Workshop: DockerFile

- Case for POSTMAN

http://192.168.99.103 | http://192.168.99.103 | http://192.168.99.103 | No Environment

POST http://192.168.99.103/users/insertuser Params Send Save

Authorization Headers (1) Body Pre-request Script Tests Code

form-data x-www-form-urlencoded raw binary JSON (application/json)

```
1 {"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"Slave"}
2
```

Body Cookies Headers (5) Tests Status: 200 OK Time: 25 ms

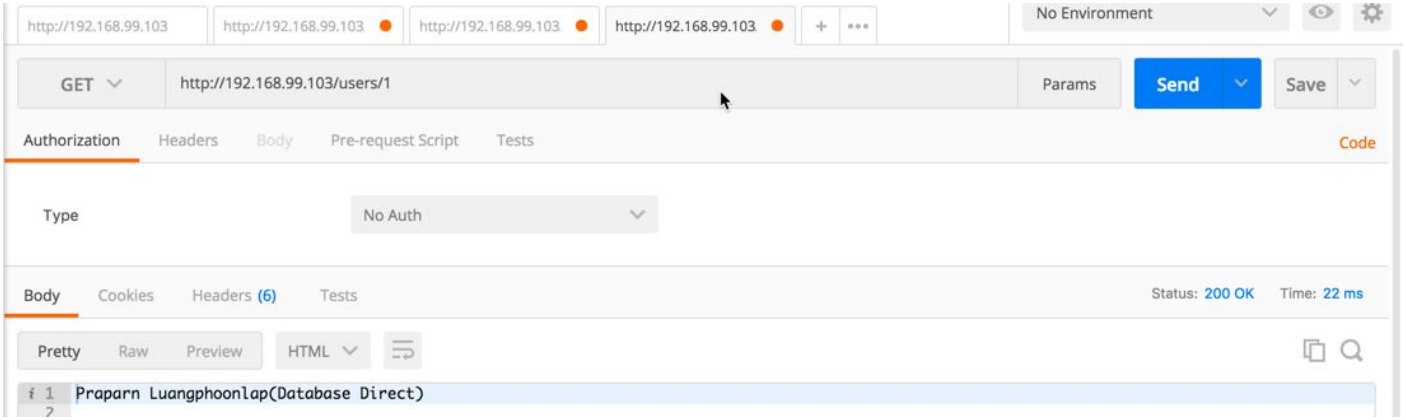
Pretty Raw Preview HTML

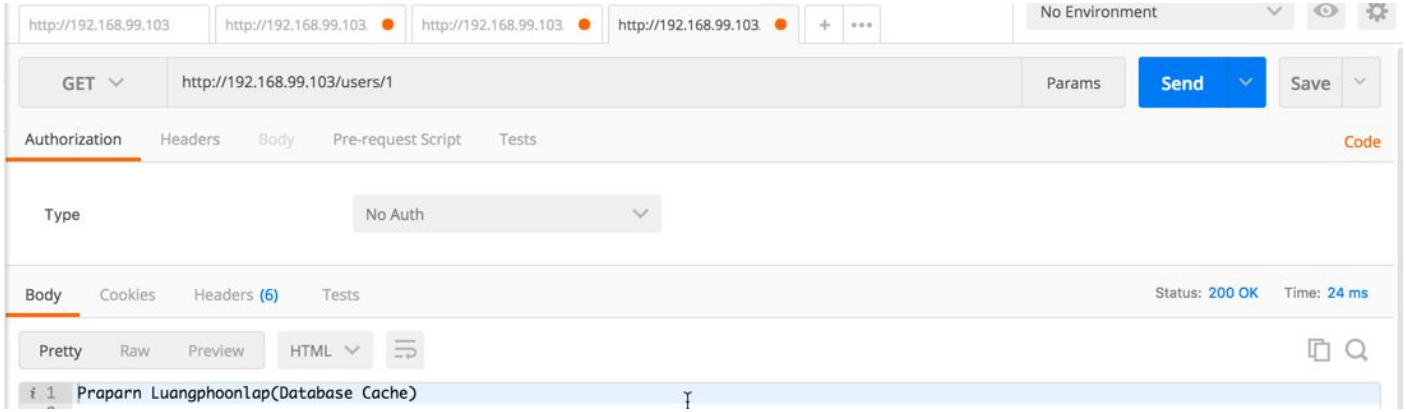
```
1 ########### Record was added ###########
2
```


kubernetes by Google

Workshop: DockerFile

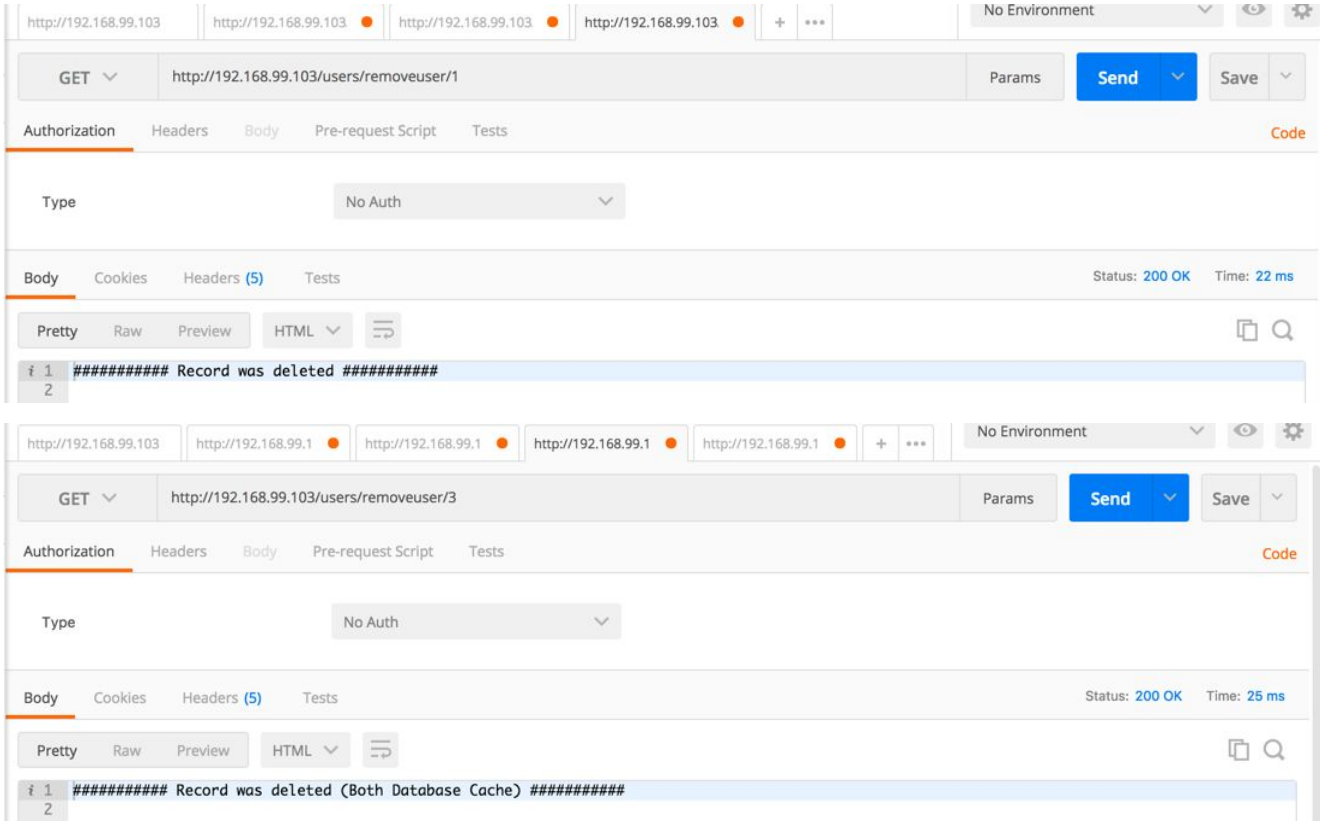
- Case for POSTMAN





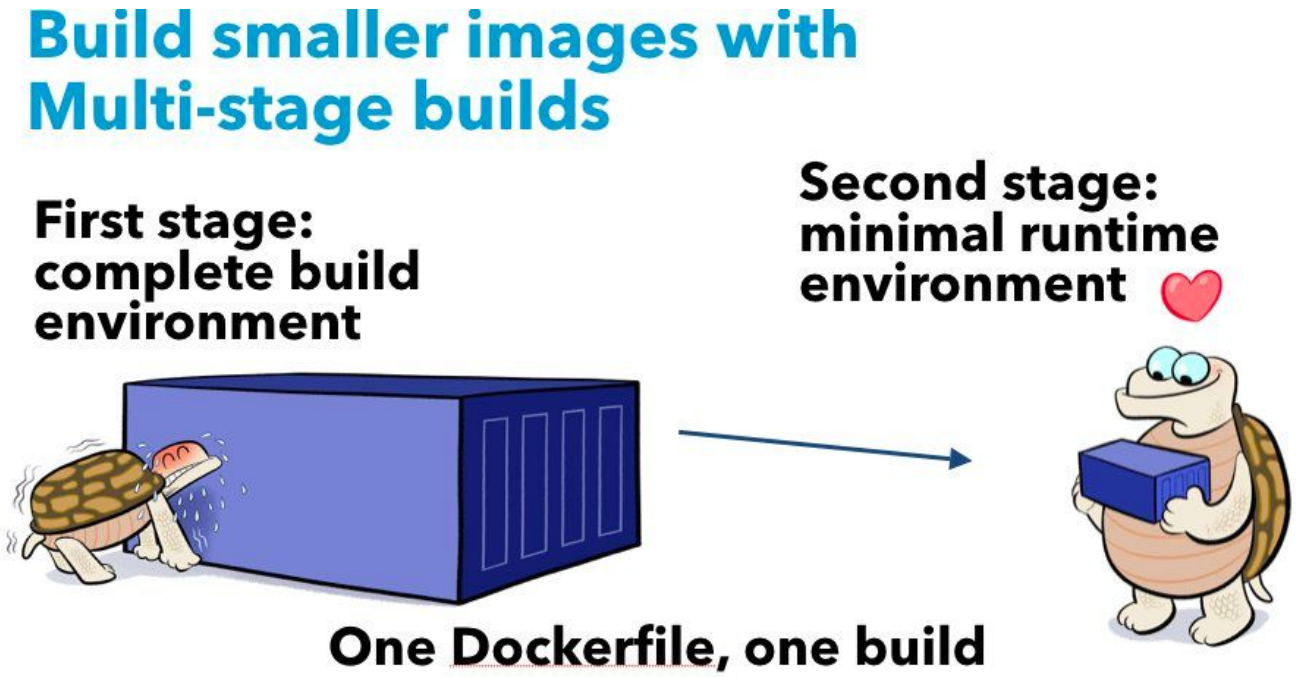
Workshop: DockerFile

- Case for POSTMAN



Dockerfile

- Multi-State was introduced on April 2017(Since 17.06)
- Write multiple “FROM” on same dockerfile
- Reduce duplicate workload by copy “intermediate state” during build to new image



Dockerfile

Multi-state dockerfile:

```
1   ############################
2   # STEP 1 build executable binary
3   ############################
4   FROM golang:alpine AS builder
5   # Install git.
6   # Git is required for fetching the dependencies.
7   RUN apk update && apk add --no-cache git
8   WORKDIR $GOPATH/src/mypackage/myapp/
9   COPY . .
10  # Fetch dependencies.
11  # Using go get.
12  RUN go get -d -v
13  # Build the binary.
14  RUN go build -o /go/bin/hello
15  ############################
16  # STEP 2 build a small image
17  ############################
18  FROM scratch
19  # Copy our static executable.
20  COPY --from=builder /go/bin/hello /go/bin/hello
21  # Run the hello binary.
22  ENTRYPOINT ["/go/bin/hello"]
```

Output: Final Image 1 Unit

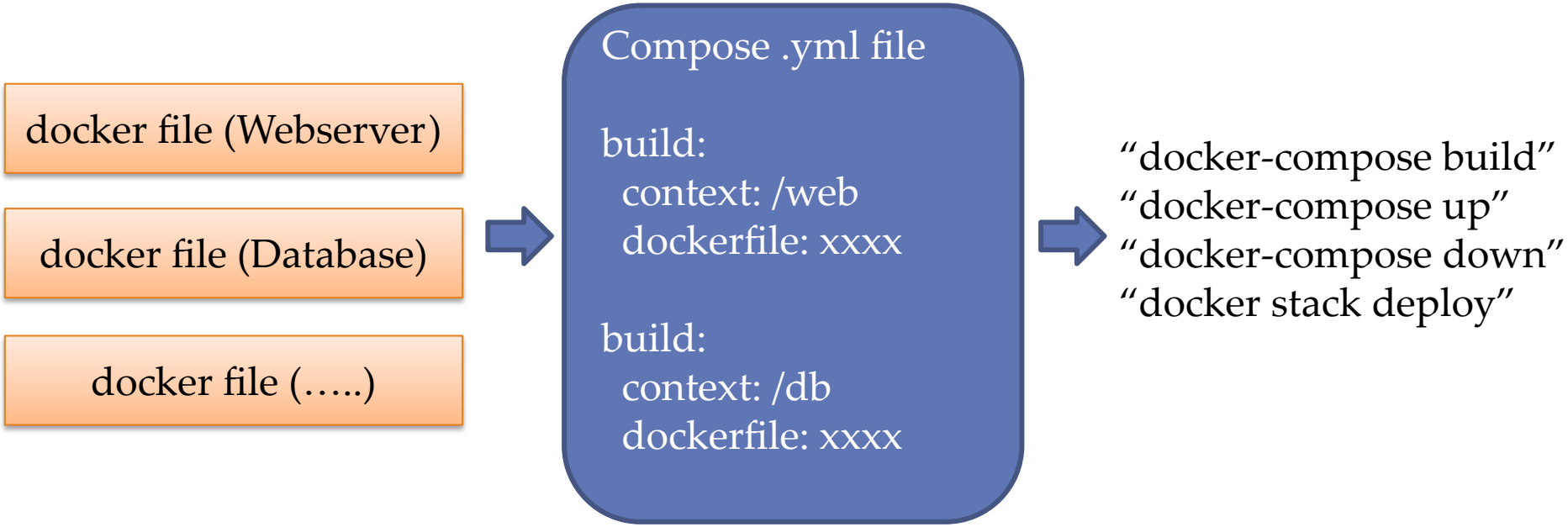
Compose

Compose

- Compose is multiple-container tools in application stack
  - Web component service
  - Application component service
  - Database component service
  - etc
- Compose can start / stop / monitor entire application stack
- Fit for application stack for minimize complicate to deploy application
- *No Support --ip now

Compose

- Step docker-compose
  - Create dockerfile for each component
  - Create compose.yml/.yaml file and reference to each dockerfile
  - docker-compose up



Compose

Compose V2

Estimated reading time: 6 minutes

Compose V2 and the new `docker compose` command

> **Important**
>
> The new Compose V2, which supports the `compose` command as part of the Docker CLI, is now available.
>
> Compose V2 integrates compose functions into the Docker platform, continuing to support most of the previous `docker-compose` features and flags. You can test the Compose V2 by simply replacing the dash (`-`) with a space, and by running `docker compose`, instead of `docker-compose`.

Starting with Docker Desktop 3.4.0, you can run Compose V2 commands without modifying your invocations, by enabling the drop-in replacement of the previous `docker-compose` with the new command. See the section Installing Compose V2 for detailed instructions how to enable the drop-in replacement.

Context of Docker Compose evolution

Introduction of the Compose specification makes a clean distinction between the Compose YAML file model and the `docker-compose` implementation. Making this change has enabled a number of enhancements, including adding the `compose` command directly into the Docker CLI, being able to "up" a Compose application on cloud platforms by simply switching the Docker context, and launching of Amazon ECS and Microsoft ACI. As the Compose specification evolves, new features land faster in the Docker CLI.

While `docker-compose` is still supported and maintained, Compose V2 implementation relies directly on the compose-go bindings which are maintained as part of the specification. This allows us to include community proposals, experimental implementations by the Docker CLI and/or Engine, and deliver features faster to users. Compose V2 also supports some of the newer additions to the Compose specification, such as profiles and GPU devices.

Context of Docker Compose evolution

Introduction of the Compose specification makes a clean distinction between the Compose YAML file model and the `docker-compose` implementation. Making this change has enabled a number of enhancements, including adding the `compose` command directly into the Docker CLI, being able to "up" a Compose application on cloud platforms by simply switching the Docker context, and launching of Amazon ECS and Microsoft ACI. As the Compose specification evolves, new features land faster in the Docker CLI.

While `docker-compose` is still supported and maintained, Compose V2 implementation relies directly on the compose-go bindings which are maintained as part of the specification. This allows us to include community proposals, experimental implementations by the Docker CLI and/or Engine, and deliver features faster to users. Compose V2 also supports some of the newer additions to the Compose specification, such as profiles and GPU devices.

Additionally, Compose V2 also supports Apple silicon.

For more information about the flags that are supported in the new compose command, see the docker-compose compatibility list.

Transitioning to GA for Compose V2

We are currently working towards providing a standard way to install Compose V2 on Linux. When this is available, Compose V2 will be marked as Generally Available (GA).

Compose V2 GA means:

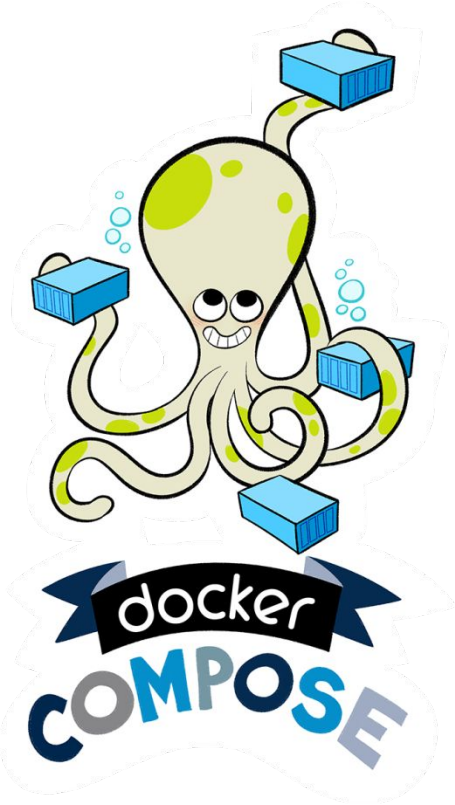
- New features and bug fixes will only be considered in the Compose V2 code base.
- Docker Compose V2 will be the default setting in Docker Desktop for Mac and Windows. You can still opt out through the Docker Desktop UI and the CLI. This means, when you run `docker-compose`, you will actually be running `docker compose`.
- Compose V2 will be included with the latest version of the Docker CLI. You can use Compose Switch to redirect `docker-compose` to `docker compose`.
- Compose V2 branch will become the default branch.
- Docker Compose V1 will be maintained to address any security issues.

Compose

- Compose syntax: (docker compose up/down/run)
  - version: x (Current Version 3)
  - services:
    - XXX (service name):
      - image: <image name>
      - build:
        - target: /xxxx (New on Version 3.4)
        - cache_from: (New on Version 3.2)
          - <image name>
        - labels: (New on Version 3.3)
          - <label>: <value>
        - context: ./<path>
        - dockerfile: <file>
      - container_name: <name>
      - args:
        - <xxxx>:<value>
      - dns: x.x.x.x
      - dns_search: xxx.com
      - entrypoint: ["nginx","-c","/etc/nginx/nginx.conf"]
      - env_file: xxxx.xxx (VAR=VAL)
      - Environment:
        - TOKEN: XXXX
      - ports:
        - - “9999:9999”
      - Link:
        - xx:<alias>
      - healthcheck:
        - test: ["CMD-SHELL", "pg_isready"]
        - interval: 30s
        - timeout: 30s
        - retries: 3
        - start_period: 100s (New on Version 3.4)

Compose

- Compose syntax:
  - XXX (service name):
    - depends_on:
      - XXXX
      - XXXX
    - networks:
      - XXXX1
      - XXXX2



Ref:https://docs.docker.com/compose/compose-file/#command

Compose

Control startup and shutdown order in Compose

Estimated reading time: 2 minutes

You can control the order of service startup and shutdown with the depends_on option. Compose always starts and stops containers in dependency order, where dependencies are determined by `depends_on`, `links`, `volumes_from`, and `network_mode: "service:..."`.

However, for startup Compose does not wait until a container is "ready" (whatever that means for your particular application) - only until it's running. There's a good reason for this.

The problem of waiting for a database (for example) to be ready is really just a subset of a much larger problem of distributed systems. In production, your database could become unavailable or move hosts at any time. Your application needs to be resilient to these types of failures.

To handle this, design your application to attempt to re-establish a connection to the database after a failure. If the application retries the connection, it can eventually connect to the database.

The best solution is to perform this check in your application code, both at startup and whenever a connection is lost for any reason. However, if you don't need this level of resilience, you can work around the problem with a wrapper script:

- Use a tool such as wait-for-it, dockerize, or sh-compatible wait-for. These are small wrapper scripts which you can include in your application's image to poll a given host and port until it's accepting TCP connections.

  For example, to use `wait-for-it.sh` or `wait-for` to wrap your service's command:

```
version: "2"
services:
  web:
    build: .
    ports:
      - "80:8000"
    depends_on:
      - "db"
    command: ["./wait-for-it.sh", "db:5432", "--", "python", "app.py"]
  db:
    image: postgres
```

Ref: https://docs.docker.com/compose/startup-order/

Compose

eficode / wait-for
forked from mrako/wait-for

Watch 20 | Star 580 | Fork 161

Code | Pull requests 12 | Projects 0 | Wiki | Security | Insights

./wait-for is a script to wait for another service to become available.

15 commits | 1 branch | 0 releases | 5 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

This branch is 8 commits ahead of mrako:master. #16 Compare

suonto Merge pull request #7 from szTheory/patch-1 ... Latest commit 8283864 on Apr 9, 2018

.gitignore	Add testing using bats	2 years ago
.travis.yml	Move travis build and status under Eficode	2 years ago
Dockerfile	Add testing using bats	2 years ago
LICENSE	initial commit	2 years ago
README.md	Fix also the other url that had capital E	last year
package.json	add name and version so it can be installed with npm	2 years ago
wait-for	use 'exec "$@"'	2 years ago
wait-for.bats	Add testing using bats	2 years ago

README.md

Wait for another service to become available

`./wait-for` is a script designed to synchronize services like docker containers. It is sh and alpine compatible. It was inspired by vishnubob/wait-for-it, but the core has been rewritten at Eficode by dsuni and mrako.

When using this tool, you only need to pick the `wait-for` file as part of your project.

build passing

Ref: https://github.com/Eficode/wait-for

Compose

vishnubob / wait-for-it

Watch 70 | Star 3,600 | Fork 910

Code | Issues 16 | Pull requests 18 | Projects 0 | Wiki | Security | Insights

Pure bash script to test and wait on the availability of a TCP host and port

36 commits | 1 branch | 0 releases | 8 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

douglas-gibbons Merge branch 'fwoelffel-master' — Latest commit 54d1f0b on Nov 4, 2018

test	Fixes to test script for flake8	2 years ago
.gitignore	Start of test framework	2 years ago
.travis.yml	Fixes to test script for flake8	2 years ago
LICENSE	license added	3 years ago
README.md	README: community section + mention Debian package	10 months ago
wait-for-it.sh	Merge branch 'master' of https://github.com/fwoelffel/wait-for-it int...	8 months ago

README.md

wait-for-it

`wait-for-it.sh` is a pure bash script that will wait on the availability of a host and TCP port. It is useful for synchronizing the spin-up of interdependent services, such as linked docker containers. Since it is a pure bash script, it does not have any external dependencies.

Usage

```
wait-for-it.sh host:port [-s] [-t timeout] [-- command args]
-h HOST | --host=HOST       Host or IP under test
-p PORT | --port=PORT       TCP port under test
                            Alternatively, you specify the host and port as host:port
-s | --strict               Only execute subcommand if the test succeeds
-q | --quiet                Don't output any status messages
-t TIMEOUT | --timeout=TIMEOUT
                            Timeout in seconds, zero for no timeout
-- COMMAND ARGS             Execute command with args after the test finishes
```

Ref: https://github.com/vishnubob/wait-for-it

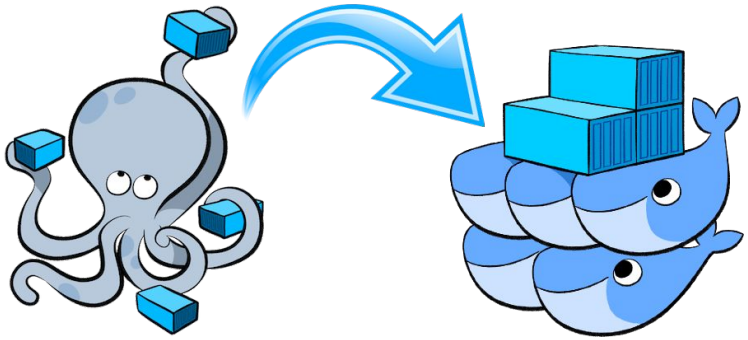
Compose

Workshop-2-1-DockerFile ▸ python_restfulset ▸ main.py

```python
from flask import Flask
from flask import Response
from flask import request
from redis import Redis
from datetime import datetime
import MySQLdb
import sys
import redis
import time
import hashlib
import os
import json
flagloop= 0
app = Flask(__name__)
startTime = datetime.now()
while flagloop == 0:
    try:
        db = MySQLdb.connect("maindb","root","password")
    except:
        time.sleep(2)
        continue
    else:
        flagloop= 1

CACHE_DB = redis.Redis(host=os.environ.get('REDIS_HOST', 'cachedb'), port=6379)
MAIN_DB = db.cursor()
@app.route('/')
def hello():
    return '<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: ' + time.strftime("%c") +'\n'
@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  ###########\n"
```

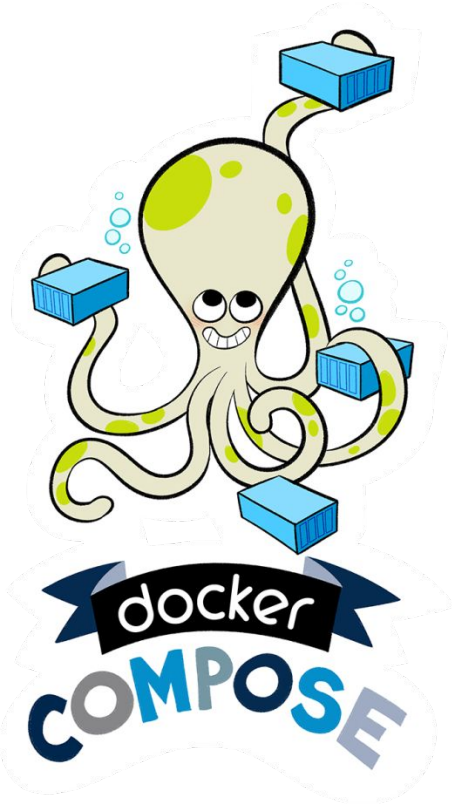
Compose

- Compose syntax: (docker stack deploy)
  - services:
    - XXX (service name):
      - image: <image name>
      - **deploy: <SWARM>**
        - mode:
          - global
          - replicated
        - resource:
          - limits:
            - cpus: '0.5'
            - memory: 100M
          - reservations:
            - cpus: '0.1'
            - memory: 20M
        - restart_policy:
          - condition: on-failure
          - delay: 5s
          - max_attempts: 3
        - update_config:
          - parallelism: 2
          - delay: 10s
          - failure_action: 10ms
          - max_failure_ratio: X
          - order: (stop-first,start-first) <New on Version 3.4>



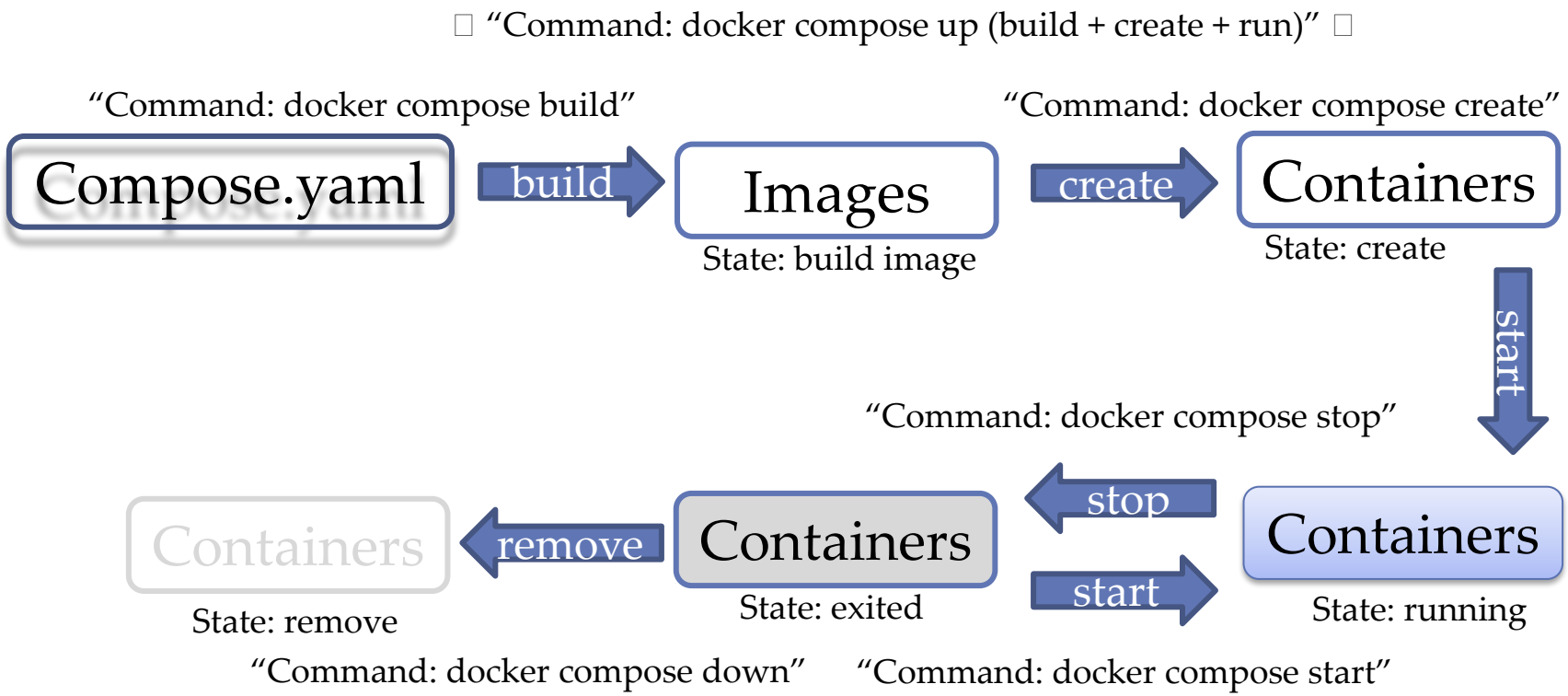
Compose

- DOCKER STACK DEPLOY
  - Not Supported
    - build
    - cgroup_parent
    - container_name
    - devices
    - dns
    - dns_search
    - external_links
    - links
    - network_mode
    - security_opt
    - stop_signal
    - sysctls
    - userns_mode



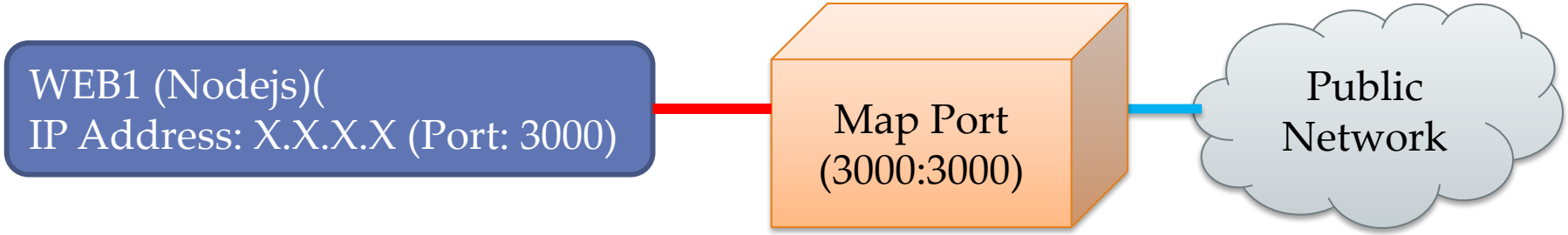
Compose

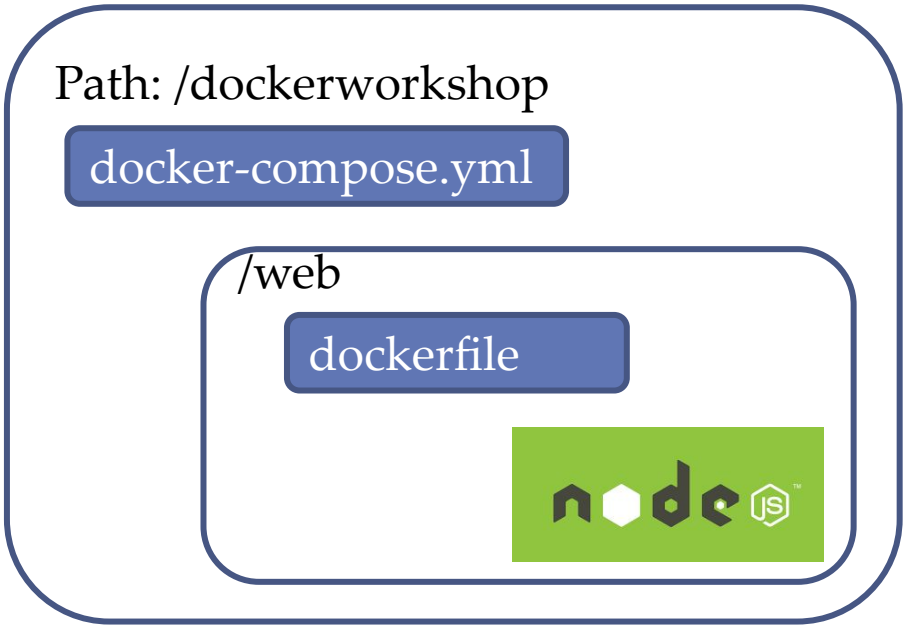
- Compose State:



Workshop: Compose

- Part1: composenodejs





Workshop: Compose

- dockerfile

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
RUN apk update && \
    apk add nodejs curl --no-cache && rm -rf /var/cache/apk/*
RUN mkdir /nodejs
COPY hello.js /nodejs/
CMD ["nginx","-c","/etc/nginx/nginx.conf"]
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:3000/ || exit 1
```

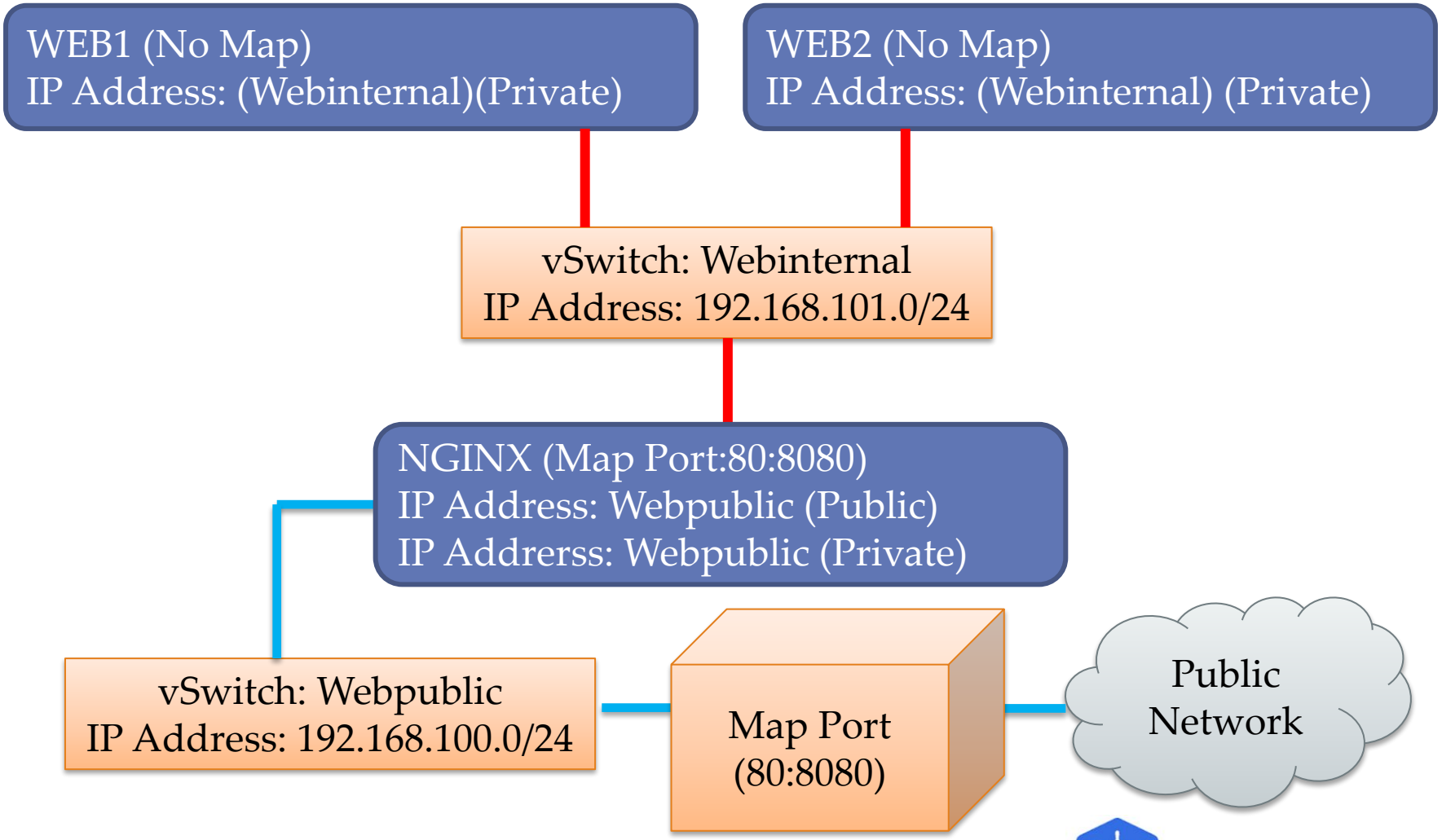
Workshop: Compose

- docker-compose.yml

```yaml
version: "3.4"
services:
  web:
    build:
      context: ./web
      dockerfile: dockerfile          # optional for specific dockerfile
    container_name: nodejs            # optional for specific container name
    command: ["node","hello.js"]      # optional for override command on dockerfile
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:3000"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                         # interval health check
      timeout: 10s                                          # timeout for check
      retries: 3                                            # maximum retries
      start_period: 30s                                     # start period <news on 3.4>
    ports:
      - 3000:3000
```

Workshop: Compose

- Part 2: composemultinodejs



Workshop: Compose

- Use case: multinodejs with nginx

Path: /dockerworkshop

docker-compose.yml

/nginx

dockerfile
(webinternal)
(webpublic)

nginx.conf
load balance:
web1
web2

/web1

dockerfile (webinternal)

/web2

dockerfile (webinternal)

Workshop: Compose

- nginx: dockerfile

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v10.14.2-r0 NPM_VERSION=v10.14.2-r0
RUN mkdir -p /run/nginx
RUN apk update && \
    apk add nginx curl --no-cache && \
    rm /etc/nginx/conf.d/default.conf \
    rm -rf /var/cache/apk/*
COPY nginx.conf /etc/nginx/nginx.conf
COPY labdocker.com.crt /etc/nginx/labdocker.com.crt
COPY labdocker.com.key /etc/nginx/labdocker.com.key
WORKDIR /usr/sbin
ENTRYPOINT ["nginx","-c","/etc/nginx/nginx.conf"]
EXPOSE 8080 8443
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:8080/ || exit 1
```

Workshop: Compose

- Web1 & Web 2: dockerfile

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
RUN apk update && \
    apk add nodejs curl --no-cache && rm -rf /var/cache/apk/*
RUN mkdir /nodejs
COPY hello.js /nodejs/
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:3000/ || exit 1
```

Workshop: Compose

- Web1: Hello.js

```
var http = require('http');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World Container Docker for Nodejs Node 1\n');
}).listen(3000, '0.0.0.0');

console.log('Server running at http://0.0.0.0:3000/');
```

- Web2: Hello.js

```
var http = require('http');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World Container Docker for Nodejs Node 2\n');
}).listen(3000, '0.0.0.0');

console.log('Server running at http://0.0.0.0:3000/');
```

Workshop: Compose

- docker-compose.yml

```yaml
version: '3.4'
services:
  nginx:
    build:
      context: ./nginx
      dockerfile: dockerfile          # optional for specific dockerfile
      cache_from:
        - alpine:latest
    container_name: nginx
    depends_on:
      - web1
      - web2
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:80"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                        # interval health check
      timeout: 10s                                         # timeout for check
      retries: 3                                           # maximum retries
      start_period: 30s                                    # start period <news on 3.4>
    networks:
      webpublic:
        aliases:
        - nginx
      webinternal:
        aliases:
        - nginx
    ports:
      - "80:8080"
```

Workshop: Compose

- docker-compose.yml

```yaml
  web1:
    build:
      context: ./web1
      dockerfile: dockerfile          # optional for specific dockerfile
      cache_from:
        - labdocker/alpineweb:latest
    container_name: web1
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:3000"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                         # interval health check
      timeout: 10s                                          # timeout for check
      retries: 3                                            # maximum retries
      start_period: 30s                                     # start period <news on 3.4>
    networks:
      webinternal:
        aliases:
        - web1

  web2:
    build:
      context: ./web2
      dockerfile: dockerfile          # optional for specific dockerfile
      cache_from:
        - labdocker/alpineweb:latest
    container_name: web2
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:3000"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                         # interval health check
      timeout: 10s                                          # timeout for check
      retries: 3                                            # maximum retries
      start_period: 30s                                     # start period <news on 3.4>
    networks:
      webinternal:
        aliases:
        - web2
```

Workshop: Compose

- docker-compose.yml

```yaml
networks:
    webpublic:
      driver: bridge
      ipam:
        driver: default
        config:
        - subnet: 192.168.100.0/24
    webinternal:
      driver: bridge
      ipam:
        driver: default
        config:
        - subnet: 192.168.101.0/24
```

Docker Security

Docker Security

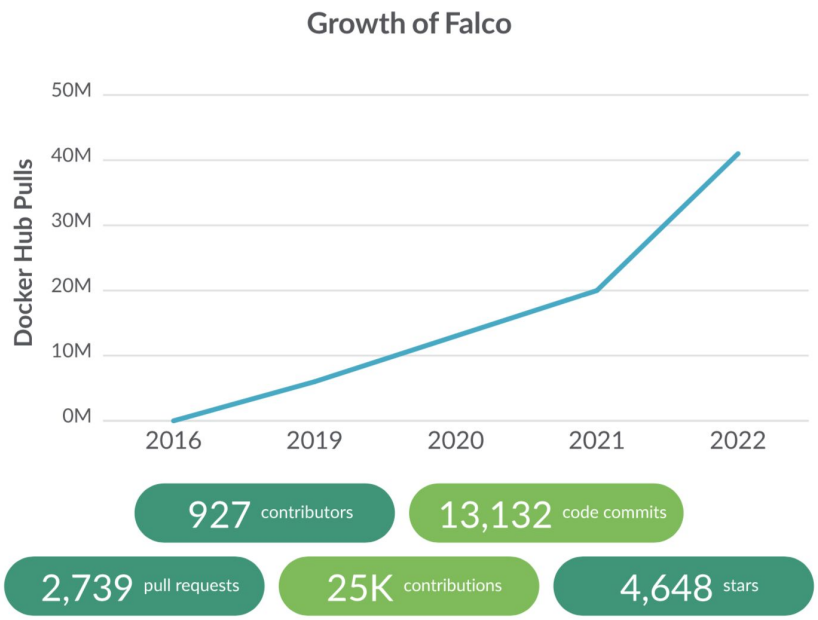
- Security on Docker
- Control Group (CG Group) / Name Space
  - Docker container run in separate namespace (Process independence, Not release / touch with other container)
  - Separate network stack
  - CG group will create for separate resource (CPU, Memory, I/O) and limit for protect single container run process and make host fail (denied-of-service)
- Docker daemon operation (root privilege)
  - Beware for allow user who operate with docker daemon
  - Some operation quite risk for security (docker pull, docker run –v with share host path etc)
  - Docker API should be aware for RESTFUL connection with secure method

Docker Security

- Secure enhancement by limit capability of container inside
  - Normally user will be “root” inside container
  - Almost container job (web server, restful server, database server) don’t need high privilege as “root” for operate
  - Control user inside container with limit capability
  - Some activity should be prohibited on container
    - Mount disk operation
    - Access system path (/bin, /usr/bin, /proc etc)
    - Create network socket
    - Execute shell
    - etc

Docker Security

- Docker Image Scanning
  - How can you make sure that your image is not vulnerability



Docker Security

- Docker Bench for Security
  - Docker container / script for check best practice to use docker in production (Ref: CIS Test)

README.md

Docker Bench for Security

```
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.5
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Nov  5 10:27:42 UTC 2019


[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 19.03.4, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:998:
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[WARN] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[WARN] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[INFO] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO]         * File not found
[WARN] 1.2.11  - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found


[INFO] 2 - Docker daemon configuration
[WARN] 2.1  - Ensure network traffic is restricted between containers on the default bridge
[PASS] 2.2  - Ensure the logging level is set to 'info'
[PASS] 2.3  - Ensure Docker is allowed to make changes to iptables
```

The Docker Bench for Security is a script that checks for dozens of common best-practices around deploying Docker containers in production. The tests are all automated, and are inspired by the CIS Docker Benchmark v1.2.0.

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     docker-bench-security
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.5
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Mar  3 12:24:34 UTC 2020


[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[INFO]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11  - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

Ref: https://github.com/docker/docker-bench-security

Docker Security

- Synk for docker image scanning (built-in)
  - Scan in each layer of docker image and compare with know vulnerability

```
ubuntu@ip-10-0-1-185:~$ docker scan labdockerthailand/alpineweb:latest

Testing labdockerthailand/alpineweb:latest...

✗ Low severity vulnerability found in nodejs/nodejs
  Description: CVE-2021-22960
  Info: https://snyk.io/vuln/SNYK-ALPINE313-NODEJS-1731455
  Introduced through: nodejs/nodejs@14.17.6-r0, nodejs/npm@14.17.6-r0
  From: nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0 > nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 14.18.1-r0

✗ Low severity vulnerability found in nodejs/nodejs
  Description: CVE-2021-22959
  Info: https://snyk.io/vuln/SNYK-ALPINE313-NODEJS-1731456
  Introduced through: nodejs/nodejs@14.17.6-r0, nodejs/npm@14.17.6-r0
  From: nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0 > nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 14.18.1-r0

✗ Medium severity vulnerability found in curl/libcurl
  Description: Insufficient Verification of Data Authenticity
  Info: https://snyk.io/vuln/SNYK-ALPINE313-CURL-1585257
  Introduced through: curl/libcurl@7.78.0-r0, curl/curl@7.78.0-r0
  From: curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0 > curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 7.79.0-r0

✗ High severity vulnerability found in curl/libcurl
  Description: Cleartext Transmission of Sensitive Information
  Info: https://snyk.io/vuln/SNYK-ALPINE313-CURL-1585248
  Introduced through: curl/libcurl@7.78.0-r0, curl/curl@7.78.0-r0
  From: curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0 > curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 7.79.0-r0

✗ Critical severity vulnerability found in curl/libcurl
  Description: Double Free
  Info: https://snyk.io/vuln/SNYK-ALPINE313-CURL-1585246
  Introduced through: curl/libcurl@7.78.0-r0, curl/curl@7.78.0-r0
  From: curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0 > curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 7.79.0-r0


Package manager:   apk
Project name:      docker-image|labdockerthailand/alpineweb
Docker image:      labdockerthailand/alpineweb:latest
Platform:          linux/amd64
Base image:        alpine:3.13.6

Tested 24 dependencies for known vulnerabilities, found 5 vulnerabilities.

According to our scan, you are currently using the most secure version of the selected base image

For more free scans that keep your images secure, sign up to Snyk at https://dockr.ly/3ePqVcp

ubuntu@ip-10-0-1-185:~$
```

Docker Security

- Default apparmor in docker (1.13.0 and above) apply on all docker contaiener

```
ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestdefault sh
/ # touch /tmp/testcreatefile
/ # ls -hl /tmp
total 0
-rw-r--r--    1 root     root           0 Mar  3 07:33 testcreatefile
/ # touch /proc/testprohibitfile
touch: /proc/testprohibitfile: No such file or directory
/ # ls -hl /proc
total 0
dr-xr-xr-x    9 root     root           0 Mar  3 07:33 1
dr-xr-xr-x    9 root     root           0 Mar  3 07:34 14
dr-xr-xr-x    9 root     root           0 Mar  3 07:33 6
drwxrwxrwt    2 root     root          40 Mar  3 07:33 acpi
-r--r--r--    1 root     root           0 Mar  3 07:34 buddyinfo
dr-xr-xr-x    4 root     root           0 Mar  3 07:33 bus
-r--r--r--    1 root     root           0 Mar  3 07:34 cgroups
-r--r--r--    1 root     root           0 Mar  3 07:34 cmdline
-r--r--r--    1 root     root           0 Mar  3 07:34 consoles
-r--r--r--    1 root     root           0 Mar  3 07:34 cpuinfo
-r--r--r--    1 root     root           0 Mar  3 07:34 crypto
-r--r--r--    1 root     root           0 Mar  3 07:34 devices
-r--r--r--    1 root     root           0 Mar  3 07:34 diskstats
-r--r--r--    1 root     root           0 Mar  3 07:34 dma
dr-xr-xr-x    2 root     root           0 Mar  3 07:34 driver
-r--r--r--    1 root     root           0 Mar  3 07:34 execdomains
-r--r--r--    1 root     root           0 Mar  3 07:34 fb
-r--r--r--    1 root     root           0 Mar  3 07:34 filesystems
dr-xr-xr-x    6 root     root           0 Mar  3 07:33 fs
-r--r--r--    1 root     root           0 Mar  3 07:34 interrupts
-r--r--r--    1 root     root           0 Mar  3 07:34 iomem
-r--r--r--    1 root     root           0 Mar  3 07:34 ioports
dr-xr-xr-x   23 root     root           0 Mar  3 07:33 irq
-r--r--r--    1 root     root           0 Mar  3 07:34 kallsyms
crw-rw-rw-    1 root     root        1,   3 Mar  3 07:33 kcore
-r--r--r--    1 root     root           0 Mar  3 07:34 key-users
crw-rw-rw-    1 root     root        1,   3 Mar  3 07:33 keys
/ # ls -hl /proc | grep test
/ # cp /tmp/testcreatefile /proc/testprohibitfile
cp: can't create '/proc/testprohibitfile': No such file or directory
/ # ls -hl /proc | grep test
/ # exit
ubuntu@ip-10-0-1-104:~$ docker container stop sectestdefault
sectestdefault
```

moby / moby

Watch 3,287 | Star 52,368 | Fork 15,155

Code | Issues 3,434 | Pull requests 183 | Projects 2 | Wiki | Insights

Branch: master | moby / profiles / apparmor / template.go | Find file | Copy path

cyphar apparmor: allow receiving of signals from 'docker kill' 4822fb1 on Sep 12, 2018

5 contributors

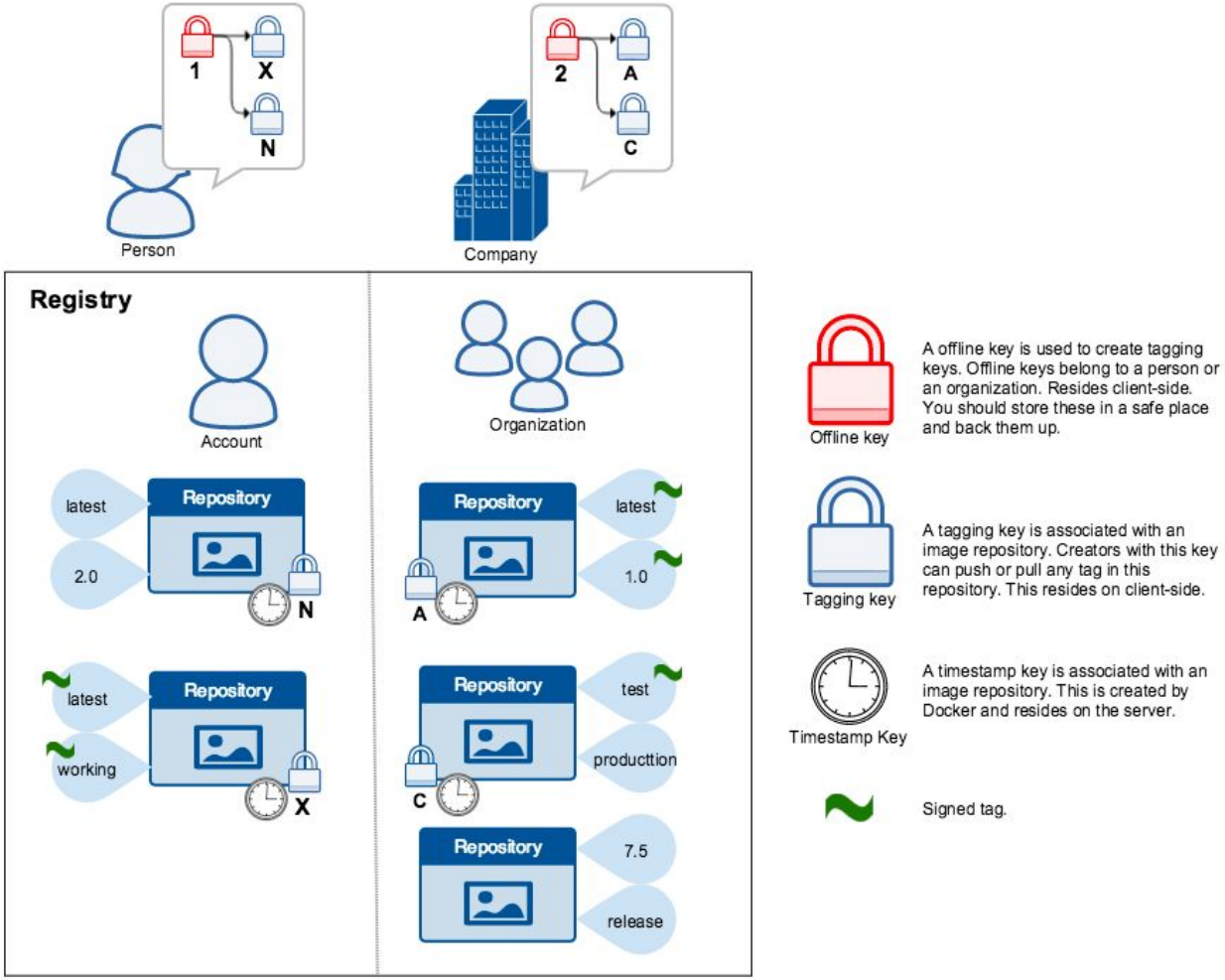
51 lines (42 sloc) | 1.53 KB | Raw | Blame | History

```
// +build linux

package apparmor // import "github.com/docker/docker/profiles/apparmor"

// baseTemplate defines the default apparmor profile for containers.
const baseTemplate = `
{{range $value := .Imports}}
{{$value}}
{{end}}

profile {{.Name}} flags=(attach_disconnected,mediate_deleted) {
{{range $value := .InnerImports}}
  {{$value}}
{{end}}

  network,
  capability,
  file,
  umount,
{{if ge .Version 208096}}
{{/* Allow 'docker kill' to actually send signals to container processes. */}}
  signal (receive) peer={{.DaemonProfile}},
{{/* Allow container processes to send signals amongst themselves. */}}
  signal (send,receive) peer={{.Name}},
{{end}}

  deny @{PROC}/* w,   # deny write for all files directly in /proc (not in a subdir)
  # deny write to files not in /proc/<number>/** or /proc/sys/**
  deny @{PROC}/{[^1-9],[^1-9][^0-9],[^1-9s][^0-9y][^0-9s],[^1-9][^0-9][^0-9][^0-9]*}/** w,
  deny @{PROC}/sys/[^k]** w,  # deny /proc/sys except /proc/sys/k* (effectively /proc/sys/kernel)
  deny @{PROC}/sys/kernel/{?,??,[^s][^h][^m]**} w,  # deny everything except shm* in /proc/sys/kernel/
  deny @{PROC}/sysrq-trigger rwklx,
  deny @{PROC}/kcore rwklx,

  deny mount,

  deny /sys/[^f]*/** wklx,
  deny /sys/f[^s]*/** wklx,
  deny /sys/fs/[^c]*/** wklx,
  deny /sys/fs/c[^g]*/** wklx,
  deny /sys/fs/cg[^r]*/** wklx,
  deny /sys/firmware/** rwklx,
  deny /sys/kernel/security/** rwklx,

{{if ge .Version 208095}}
  # suppress ptrace denials when using 'docker ps' or using 'ps' inside a container
  ptrace (trace,read) peer={{.Name}},
{{end}}
}
`
```

Ref: https://github.com/moby/moby/tree/master/profiles/apparmor

Docker Security

- Image Content Trust



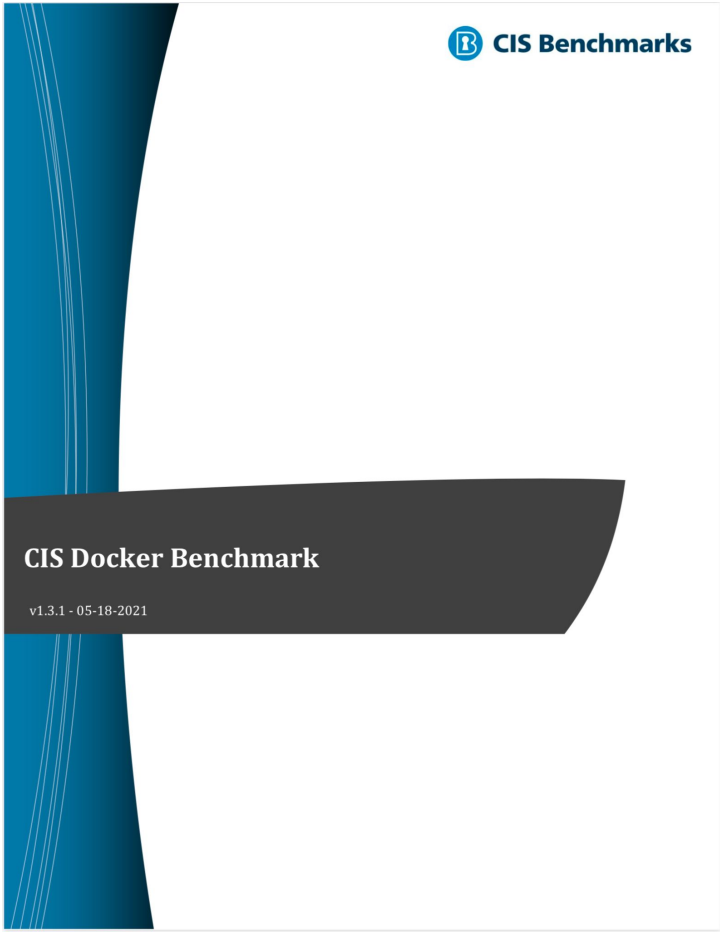
Docker Security

- Recommendation for enhance docker security
  - User who have docker privilege
  - Non-root user for manage docker daemon operation (docker run etc)
  - Image Content Trust (Sign Image) for pull/push image
  - Docker daemon socket with HTTP socket (with TLS verify and authentication)
  - Limit user on container (Secure compute mode (seccomp)

Workshop: Docker Security

- Docker Bench for Security
- Purpose: Check vulnerability on docker server on production environment

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     docker-bench-security
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.5
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Mar  3 12:24:34 UTC 2020


[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[INFO]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11  - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```



Workshop: Docker Security

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     docker-bench-security
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.5
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Mar  3 12:24:34 UTC 2020


[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[INFO]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11  - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

1.1 Linux Hosts Specific Configuration

This section contains recommendations that securing Linux Hosts running Docker Containers.

1.1.1 Ensure a separate partition for containers has been created (Automated)

Profile Applicability:

- Level 1 - Linux Host OS

Description:

All Docker containers and their data and metadata is stored under `/var/lib/docker` directory. By default, `/var/lib/docker` should be mounted under either the `/` or `/var` partitions dependent on how the Linux operating system in use is configured.

Rationale:

Docker depends on `/var/lib/docker` as the default directory where all Docker related files, including the images, are stored. This directory could fill up quickly causing both Docker and the host to become unusable. For this reason, you should create a separate partition (logical volume) for storing Docker files.

Impact:

None.

Audit:

At the Docker host execute one of the below commands:

```
grep '/var/lib/docker\s' /proc/mounts
```

This should return the partition details for the `/var/lib/docker` mountpoint.

```
mountpoint -- "$(docker info -f '{{ .DockerRootDir }}')"
```

This should return whether the configured root directory is a mount point.

Workshop: Docker Security

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     docker-bench-security
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.5
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Mar  3 12:24:34 UTC 2020


[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[INFO]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11  - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

1.6 Ensure auditing is configured for Docker files and directories - /var/lib/docker (Scored)

Profile Applicability:

- Level 1 - Linux Host OS

Description:

Audit `/var/lib/docker`.

Rationale:

Apart from auditing your regular Linux file system and system calls, audit all Docker related files and directories. Docker daemon runs with `root` privileges. Its behavior depends on some key files and directories. `/var/lib/docker` is one such directory. It holds all the information about containers. It must be audited.

Audit:

Verify that there is an audit rule corresponding to `/var/lib/docker` directory.

For example, execute below command:

```
auditctl -l | grep /var/lib/docker
```

This should list a rule for `/var/lib/docker` directory.

Remediation:

Add a rule for `/var/lib/docker` directory.

For example,

Add the line as below in `/etc/audit/audit.rules` file:

```
-w /var/lib/docker -k docker
```

Then, restart the audit daemon. For example,

```
service auditd restart
```

Workshop: Docker Security

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     docker-bench-security
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.5
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Mar  3 12:24:34 UTC 2020


[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[INFO]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11  - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

```
ubuntu@ip-10-0-1-234:~$ sudo more /etc/audit/rules.d/audit.rules
## First rule - delete all
-D

## Increase the buffers to survive stress events.
## Make this bigger for busy systems
-b 8192

## This determine how long to wait in burst of events
--backlog_wait_time 0

## Set failure mode to syslog
-f 1

-w /usr/bin/docker -p wa
-w /var/lib/docker -p wa
-w /etc/docker -p wa
-w /lib/systemd/system/docker.service -p wa
-w /lib/systemd/system/docker.socket -p wa
-w /etc/default/docker -p wa
-w /etc/docker/daemon.json -p wa
-w /usr/bin/docker-containerd -p wa
-w /usr/bin/docker-runc -p wa
-w /usr/bin/docker -k docker
ubuntu@ip-10-0-1-234:~$
```

Workshop: Docker Security

- Apparmor
- Purpose: Restrict privilege on container runtime

```
ubuntu@ip-10-0-1-104:~$ docker container run -dt --name sectestrestrict --security-opt "apparmor=restrict-apparmor" labdocker/alpine:latest sh
6db2a1be7b9d1b3f15409e49bc1a0e1a9373cc9b8740a017ff8fb163283b8d1b
ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestdefault sh
Error: No such container: sectestdefault
ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestrestrict sh
/ # ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1): 56 data bytes
ping: can't create raw socket: Permission denied
/ # apk update
ERROR: Unable to lock database: Permission denied
ERROR: Failed to open apk database: Permission denied
/ # apk add curl
ERROR: Unable to lock database: Permission denied
ERROR: Failed to open apk database: Permission denied
/ # curl https://www.google.com
sh: curl: not found
/ # touch /tmp/testcreatefile1
touch: /tmp/testcreatefile1: Permission denied
/ # touch /home/testcreatefile2
touch: /home/testcreatefile2: Permission denied
/ # touch /etc/testcreatefile3
touch: /etc/testcreatefile3: Permission denied
/ # touch /proc/testcreatefile4
touch: /proc/testcreatefile4: No such file or directory
/ # cp /tmp/testcreatefile /proc/testprohibitfile
cp: can't stat '/tmp/testcreatefile': No such file or directory
/ # ls -hl /proc | grep test
/ # exit
ubuntu@ip-10-0-1-104:~$ docker container stop sectestrestrict && docker container rm sectestrestrict
sectestrestrict
sectestrestrict
ubuntu@ip-10-0-1-104:~$
```

Workshop: Docker Security

- Content Trust
- Purpose: enable feature "DOCKER_CONTENT_TRUST" with sign/unsign

```
root@labdocker:~# docker push praparn/alpine:sign
The push refers to a repository [docker.io/praparn/alpine]
6102f0d2ad33: Layer already exists
sign: digest: sha256:65242e8220a341cec40628caaea77eb4acd2fc252329aa853526fde15a4a1d85 size: 528
Signing and pushing trust metadata
You are about to create a new root signing key passphrase. This passphrase
will be used to protect the most sensitive key in your signing system. Please
choose a long, complex passphrase and be careful to keep the password and the
key file itself secure and backed up. It is highly recommended that you use a
password manager to generate the passphrase and keep it safe. There will be no
way to recover this key. You can find the key in your config directory.
Enter passphrase for new root key with ID ca6ab4b:
Repeat passphrase for new root key with ID ca6ab4b:
Enter passphrase for new repository key with ID 68d88cd (docker.io/praparn/alpine):
Repeat passphrase for new repository key with ID 68d88cd (docker.io/praparn/alpine):
```

Registry

Registry

- Registry is type of container for keep image on private server without need to upload on cloud. Registry can keep image as safe inside organization

Ex: labdocker/alpineweb:latest

- Registry will store the image same as hub.docker.com
- Registry between machine need to authentication with TLS certification
- Storage backend have several option for use
  - Etc: s3 (aws), azure, gcs (google cloud) etc
  - option gc (garbage collect) ด้วย option
  - “bin/registry garbage-collect [--dry-run] config.yml”

Workshop: Registry

- Part1:Registry on single docker-machine
- Start registry and download image registry

```
docker image pull registry:2.8.0 (registry version 2.0)
```

- Start container with command below

```
docker container run -d -p 5000:5000 \
--restart=always --name registry \
 -e REGISTRY_STORAGE_DELETE_ENABLED=true \
--mount type=bind,source=/home/docker,target=/var/lib/registry \
registry:2.8.0
```

- tag image and push image docker to registry

```
docker image tag labdocker/alpine:3.13.6 \
localhost:5000/alpine:3.13.6
```

```
docker push localhost:5000/alpine:3.13.6
```

- *Need to record digest image

Workshop: Registry

- Rest API was provide to manage image on registry
- List image

```
curl -X GET http://localhost:5000/v2/_catalog
```

- Verity image revision

```
curl -X GET http://localhost:5000/v2/<name>/tags/list
```

```
curl -X GET http://localhost:5000/v2/alpine/tags/list
```

- Delete image with digest

```
curl -X DELETE
http://localhost:5000/v2/alpine/manifests/<digest>
```

Workshop: Registry

- Check image store on physical path

```
cd /home/docker
cd /docker/registry/v2
```

- Delete image

```
docker image rm localhost:5000/alpine:3.13.6
```

- Pull image

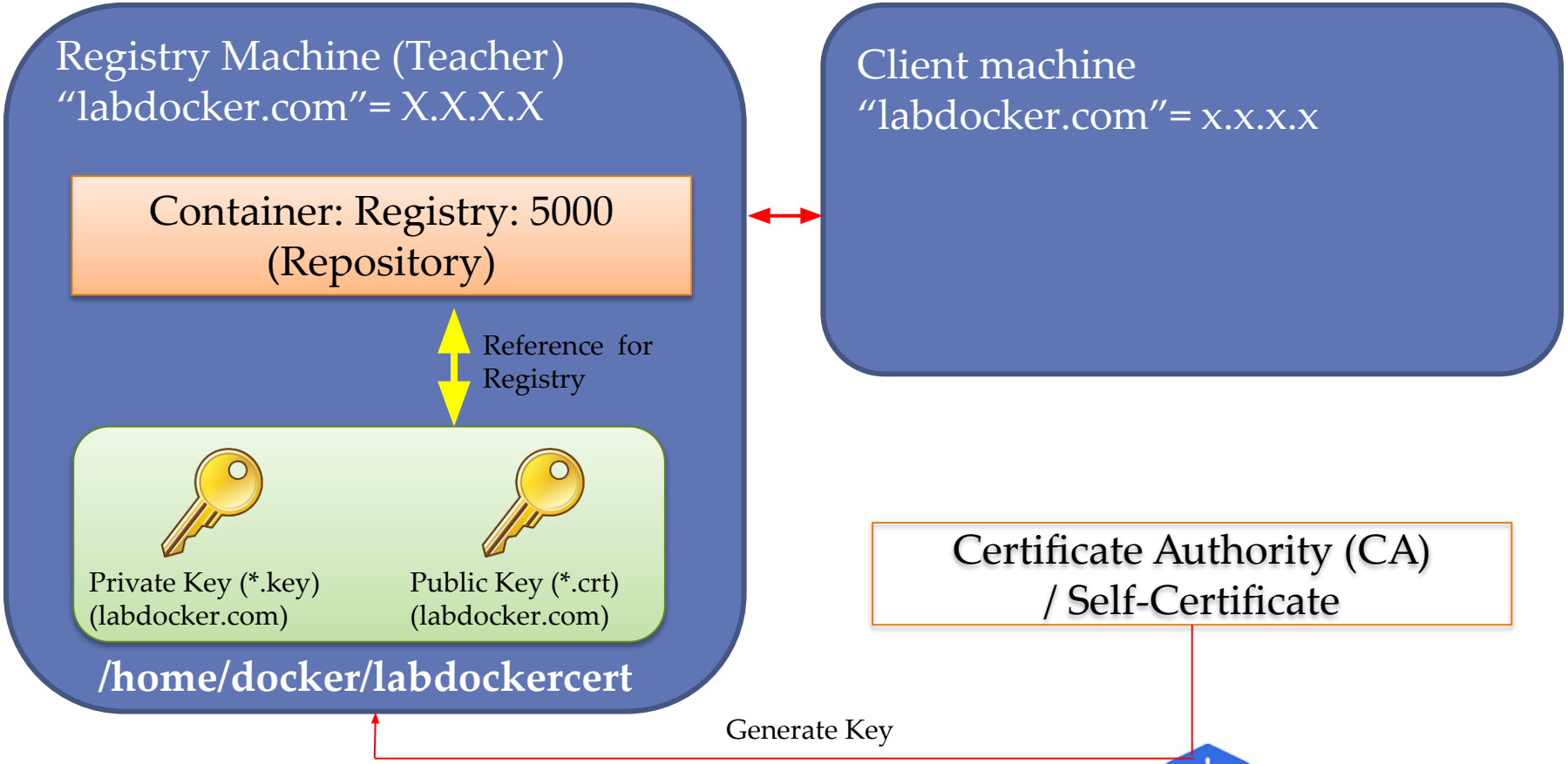
```
docker image pull localhost:5000/alpine:3.13.6
```

- Delete container and remove registry

```
docker container stop registry
docker container rm registry
```

Registry on difference machine

- Part2: Registry on multiple docker-machine
- Registry need TLS certificate (next generate of SSL) for trust registry between machine



Registry on difference machine

- Certificate from CA / Self-Signed

www.selfsignedcertificate.com

Search

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: www.example.com Generate »

sslstore
AFFORDABLE SSL CERTIFICATES
ISSUED IN MINUTES*

Registry on difference machine

- Add dns "labdocker.com" on /etc/hosts (registry, all machine)

192.168.99.100 - PuTTY

```
# The following lines are desirable for IPv6 capable hosts
# (added automatically by netbase upgrade)

::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
192.168.99.100 labdocker.com
I /etc/hosts 12/12 100%
```

192.168.99.100 - PuTTY

```
docker@labdocker:~$ sudo vi /etc/hosts
docker@labdocker:~$ ping labdocker.com
PING labdocker.com (192.168.99.100): 56 data bytes
64 bytes from 192.168.99.100: seq=0 ttl=64 time=0.110 ms
64 bytes from 192.168.99.100: seq=1 ttl=64 time=0.108 ms
64 bytes from 192.168.99.100: seq=2 ttl=64 time=0.105 ms
^C
--- labdocker.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.105/0.107/0.110 ms
docker@labdocker:~$
```

Registry on difference machine

- On ubuntu for apply with self-certificate. Need to trust root ca As below

```
sudo mkdir /usr/local/share/ca-certificates/labdocker.com:5000
sudo cp labdocker.com.crt  /usr/local/share/ca-certificates/labdocker.com:5000
sudo update-ca-certificates
sudo service docker restart
```

```
ubuntu@ip-10-0-1-182:~$ ls ~/docker-workshop-112018-cloud/Workshop-2-3-Registry
instruction.txt  labdocker.com.crt  labdocker.com.key
ubuntu@ip-10-0-1-182:~$ cd ~/docker-workshop-112018-cloud/Workshop-2-3-Registry
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo mkdir /usr/local/share/ca-certificates/labdocker.com:5000
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo cp labdocker.com.crt  /usr/local/share/ca-certificates/labdocker.com:5000
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo update-ca-certificates
Updating certificates in /etc/ssl/certs...
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo service docker restart
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$
```

Registry on difference machine

- Start container as command below

```
docker container run -d -p 5000:5000 \
        --restart=always --name registrylab \
        --mount type=bind,source=$(pwd),target=/var/lib/registry \
        --mount type=bind,source=$(pwd),target=/certs  \
        --mount type=bind,source=$(pwd),target=/auth\
        -e "REGISTRY_AUTH=htpasswd" \
        -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" \
        -e REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd \
        -e REGISTRY_STORAGE_DELETE_ENABLED=true \
        -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/labdocker.com.crt \
        -e REGISTRY_HTTP_TLS_KEY=/certs/labdocker.com.key \
        registry:2.8.0
```

- Tag image,login and test push image docker to registry

```
docker image tag labdocker/alpine:3.13.6 labdocker.com:5000/alpine:1.0
docker login -u <username> -p <password>
docker image push labdocker.com:5000/alpine:1.0
```

Registry on difference machine

- To operation registry server with TLS we need to use parameter "--cacert " and specify certificate for authentication

- List Image

```
curl –u <username:password> --cacert /Share_DockerToolbox/labdocker.com.crt   -X
GET https://labdocker.com:5000/v2/_catalog
```

- Verity image revision

```
curl –u <username:password> --cacert /home/docker/labdockercert/labdocker.com.crt
-X GET https://labdocker.com:5000/v2/alpine/tags/list
```

- Delete image with digest

```
curl –u <username:password> --cacert /home/docker/labdockercert/labdocker.com.crt
-X DELETE https://labdocker.com:5000/v2/alpine/manifests
```

Registry on difference machine

- Pull image across server

```
ubuntu@ip-10-0-1-239:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ docker login labdocker.com:5000
Username: docker
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-10-0-1-239:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ docker image pull labdocker.com:5000/alpine:1.0
1.0: Pulling from alpine
Digest: sha256:02892826401a9d18f0ea01f8a2f35d328ef039db4e1edcc45c630314a0457d5b
Status: Downloaded newer image for labdocker.com:5000/alpine:1.0
ubuntu@ip-10-0-1-239:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$
```

- Optional: Add username/password authentication

```
-v `pwd`/auth:/auth \
 -e "REGISTRY_AUTH=htpasswd" \
 -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry
Realm" \
```

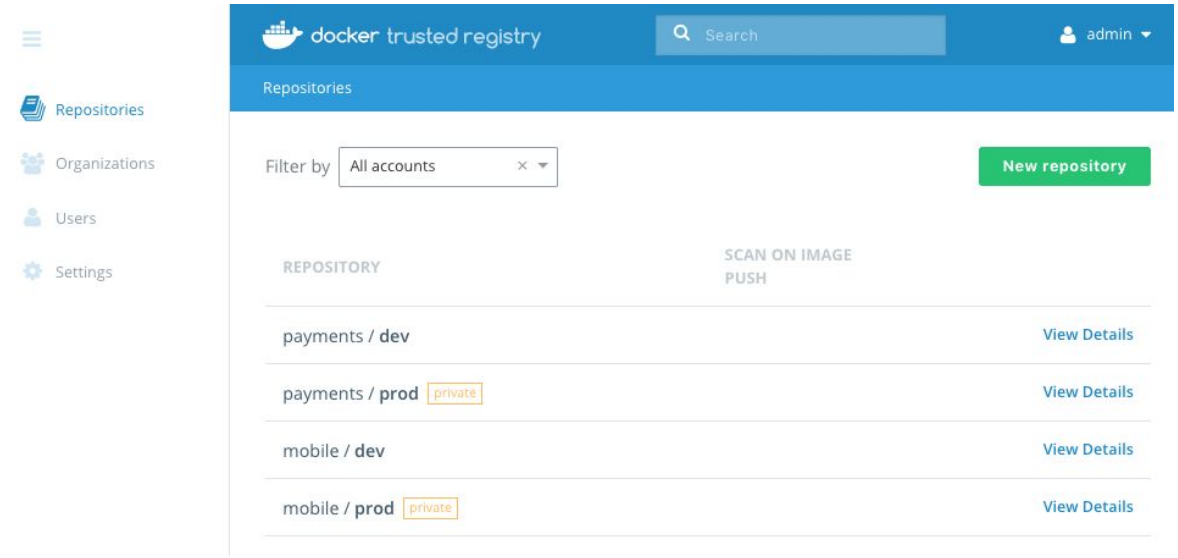
- Delete and Remove registry container

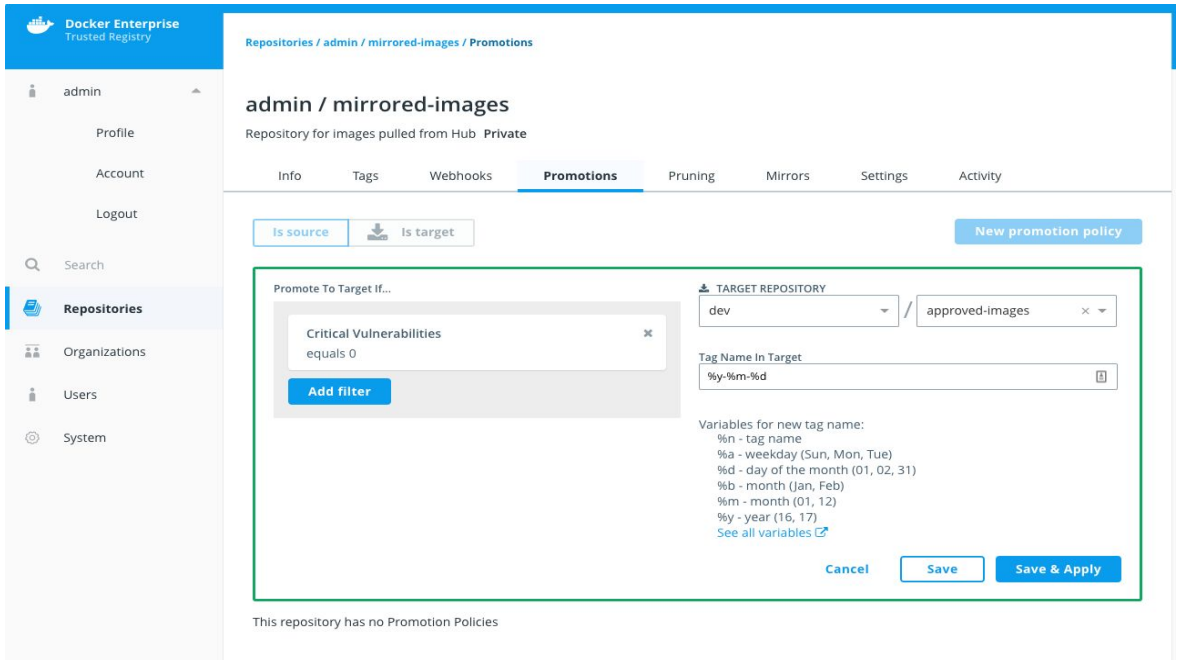
```
docker container stop registry
docker container rm -v registry
```

Ref: https://docs.docker.com/registry/deploying/

Docker Trust Registry (DTR)

- Commercial support
- GUI for manage everything
- Integrated LDAP authentication
- GC schedule available





Artifact Registry

- Sonatype

Sonatype | Documentation My Sonatype Community Guides Learn Support Who is Sonatype?

Search the docs...

Switch to another product

NEXUS REPOSITORY MANAGER 3

- Download
- Release Notes
- System Requirements
- Upgrade Compatibility - Repository Manager 2 to 3
- Repository Manager 2 to 3 Feature Equivalency
- Repository Manager Feature Matrix
- Repository Manager Concepts
- Supported Formats
- Installation
- Upgrading
- User Interface
- Configuration
- Backup and Restore
- Cleanup Policies
- High Availability
- Quick Start Guide - Proxying Maven and NPM
- Staging
- Tagging

Nexus Repository Manager 3 > Private Registry for Docker

Private Registry for Docker

Available in Nexus Repository OSS and Nexus Repository Pro

Docker containers and their usage have revolutionized the way applications and the underlying operating system are packaged and deployed to development, testing and production systems.

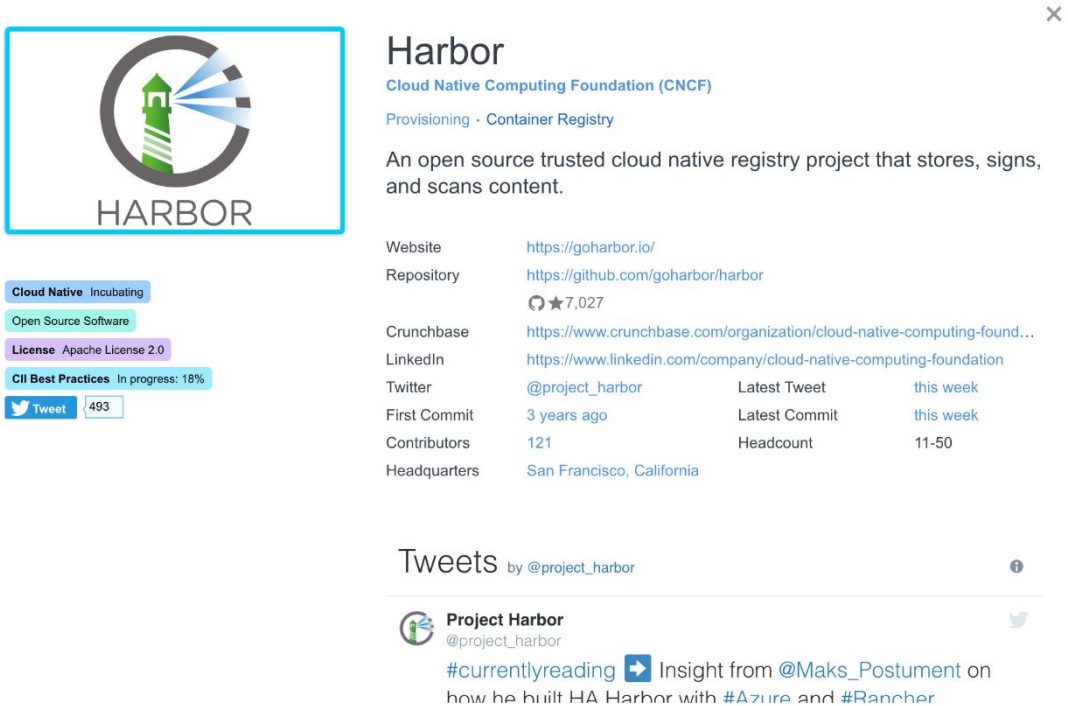
The creation of the Open Container Initiative, and the involvement of a large number of stakeholders, guarantees that the ecosystem of tools around the lightweight containers and their usage will continue to flourish.

Docker Hub is the original registry for Docker container images and it is being joined by more and more other publicly available registries such as the Google Container Registry and others.

Nexus Repository Manager Pro and Nexus Repository Manager OSS support Docker registries as the Docker repository format for hosted and proxy repositories. You can expose these repositories to the client-side tools directly or as a repository group, which is a repository that merges and exposes the contents of multiple repositories in one convenient URL. This allows you to reduce time and bandwidth usage for accessing Docker images in a registry as well as share your images within your organization in a hosted repository. Users can then launch containers based on those images, resulting in a completely private Docker registry with all the features available in the repository manager.

Artifact Registry

- Harbor CNCF Registry (Incubator)



Recap Day 2

- ***Recap Container Concept**
- Inspect/Log/Commit
- Dockerfile and Build
- Compose
- Docker Security
- Registry

Q&A

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

DAY 3

Outline Day 3

- 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)

K8S LabSheet

- https://tinyurl.com/y8yazrxy

LabSheet

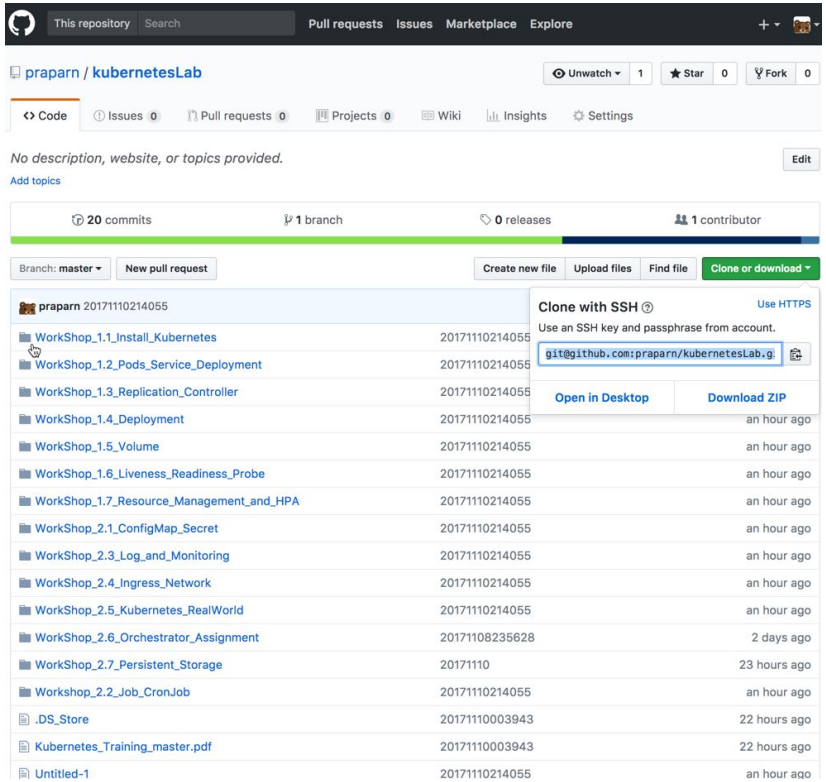
ไฟล์ แก้ไข ดู แทรก รูปแบบ ข้อมูล เครื่องมือ ส่วนเสริม ความช่วยเหลือ แก้ไขล่าสุด 1 นาทีที่ผ่านมา

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	

Lab Resource

- Download on Google Drive
  - http://tinyurl.com/yxoe8aj6
- Download on GitHub
  - Git clone https://github.com/praparn/kubernetes_202203



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 1.23

Kubernetes 1.23: The Next Frontier

Tuesday, December 07, 2021

Authors: Kubernetes 1.23 Release Team

We're pleased to announce the release of Kubernetes 1.23, the last release of 2021!

This release consists of 47 enhancements: 11 enhancements have graduated to stable, 17 enhancements are moving to beta, and 19 enhancements are entering alpha. Also, 1 feature has been deprecated.

Major Themes

Deprecation of FlexVolume

FlexVolume is deprecated. The out-of-tree CSI driver is the recommended way to write volume drivers in Kubernetes. See this doc for more information. Maintainers of FlexVolume drivers should implement a CSI driver and move users of FlexVolume to CSI. Users of FlexVolume should move their workloads to the CSI driver.

Deprecation of klog specific flags

To simplify the code base, several logging flags were marked as deprecated in Kubernetes 1.23. The code which implements them will be removed in a future release, so users of those need to start replacing the deprecated flags with some alternative solutions.

Software Supply Chain SLSA Level 1 Compliance in the Kubernetes Release Process

Kubernetes releases now generate provenance attestation files describing the staging and release phases of the release process. Artifacts are now verified as they are handed over from one phase to the next. This final piece completes the work needed to comply with Level 1 of the SLSA security framework (Supply-chain Levels for Software Artifacts).

IPv4/IPv6 Dual-stack Networking graduates to GA

IPv4/IPv6 dual-stack networking graduates to GA. Since 1.21, Kubernetes clusters have been enabled to support dual-stack networking by default. In 1.23, the `IPv6DualStack` feature gate is removed. The use of dual-stack networking is not mandatory. Although clusters are enabled to support dual-stack networking, Pods and Services continue to default to single-stack. To use dual-stack networking Kubernetes nodes must have routable IPv4/IPv6 network interfaces, a dual-stack capable CNI network plugin must be used, Pods must be configured to be dual-stack and Services must have their `.spec.ipFamilyPolicy` field set to either `PreferDualStack` or `RequireDualStack`.

HorizontalPodAutoscaler v2 graduates to GA

The HorizontalPodAutscaler `autoscaling/v2` stable API moved to GA in 1.23. The HorizontalPodAutoscaler `autoscaling/v2beta2` API has

Ref: https://kubernetes.io/blog/2021/12/07/kubernetes-1-23-release-announcement/

Kubernetes 1.23

Generic Ephemeral Volume feature graduates to GA

The generic ephemeral volume feature moved to GA in 1.23. This feature allows any existing storage driver that supports dynamic provisioning to be used as an ephemeral volume with the volume's lifecycle bound to the Pod. All StorageClass parameters for volume provisioning and all features supported with PersistentVolumeClaims are supported.

Skip Volume Ownership change graduates to GA

The feature to configure volume permission and ownership change policy for Pods moved to GA in 1.23. This allows users to skip recursive permission changes on mount and speeds up the pod start up time.

Allow CSI drivers to opt-in to volume ownership and permission change graduates to GA

The feature to allow CSI Drivers to declare support for fsGroup based permissions graduates to GA in 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.

Container Runtime Interface (CRI) v1 is default

The Kubelet now supports the CRI `v1` API, which is now the project-wide default. If a container runtime does not support the `v1` API, Kubernetes will fall back to the `v1alpha2` implementation. There is no intermediate action required by end-users, because `v1` and `v1alpha2` do not differ in their implementation. It is likely that `v1alpha2` will be removed in one of the future Kubernetes releases to be able to develop `v1`.

Structured logging graduate to Beta

Structured logging reached its Beta milestone. Most log messages from kubelet and kube-scheduler have been converted. Users are encouraged to try out JSON output or parsing of the structured text format and provide feedback on possible solutions for the open issues, such as handling of multi-line strings in log values.

Simplified Multi-point plugin configuration for scheduler

The kube-scheduler is adding a new, simplified config field for Plugins to allow multiple extension points to be enabled in one spot. The new `multiPoint` plugin field is intended to simplify most scheduler setups for administrators. Plugins that are enabled via `multiPoint` will automatically be registered for each individual extension point that they implement. For example, a plugin that implements Score and Filter extensions can be simultaneously enabled for both. This means entire plugins can be enabled and disabled without having to manually edit individual extension point settings. These extension points can now be abstracted away due to their irrelevance for most users.

Ref: https://kubernetes.io/blog/2021/12/07/kubernetes-1-23-release-announcement/

Dockershim and his story

MIRANTIS

Products Use Cases Delivery Approach BOOK A DEMO CONTACT US

Mirantis to take over support of Kubernetes dockershim

Adam Parco- December 4, 2020 - kubernetes

The rumors of dockershim's demise have been greatly exaggerated. If you follow the Kubernetes ecosystem, you may have been caught up in the ~~consternation~~ excitement over the announcement that starting with the soon-to-be-released Kubernetes 1.20, users will receive a warning that dockershim is being deprecated, and will be removed in a future release. For many people this has sparked a moment of panic, but take a deep breath, everything is going to be OK.

Even better news, however, is that Mirantis and Docker have agreed to partner to maintain the shim code standalone outside Kubernetes, as a conformant CRI interface for the Docker Engine API. For Mirantis customers, that means that Docker Engine's commercially supported version, Mirantis Container Runtime (MCR), will be CRI compliant. We will start with the great initial prototype from Dims at https://github.com/dims/cri-dockerd and continue to make it available as an open source project, https://github.com/Mirantis/cri-dockerd. This means that you can continue to build Kubernetes based on the Docker Engine as before, just switching from the built in dockershim to the external one. We will work together on making sure it continues to work as well as before and that it passes all the conformance tests and continues to work just like the built in version did. Mirantis will be using this in Mirantis Kubernetes Engine, and Docker will continue to ship this shim in Docker Desktop.

Get blogs and other content delivered straight to your inbox.

Email

Country

SUBSCRIBE

LIVE WEBINAR

Want to learn OpenStack? Join us for OpenStack Basics

Thursday March 25, 8:00 PDT

SAVE YOUR SEAT

FREE VIRTUAL TRAINING

Creating K8s objects with YAML

Ref: https://www.mirantis.com/blog/mirantis-to-take-over-support-of-kubernetes-dockershim-2/

Dockershim and his story

kubernetes

Documentation | Kubernetes Blog | Training | Partners | Community | Case Studies | Versions | English

Search

- Home
- Getting started
  - Release notes and version skew
  - Learning environment
  - **Production environment**
    - **Container runtimes**
    - Installing Kubernetes with deployment tools
    - Turnkey Cloud Solutions
    - Windows in Kubernetes

Kubernetes Documentation / Getting started / Production environment / Container runtimes

Container runtimes

You need to install a container runtime into each node in the cluster so that Pods can run there. This page outlines what is involved and describes related tasks for setting up nodes.

This page lists details for using several common container runtimes with Kubernetes, on Linux:

- containerd
- CRI-O
- Docker

Note: For other operating systems, look for documentation specific to your platform.

- Edit this page
- Create child page
- Create an issue
- Print entire section

Cgroup drivers
Container runtimes
containerd
CRI-O
Docker

containerd

This section contains the necessary steps to use containerd as CRI runtime.

Use the following commands to install Containerd on your system:

Install and configure prerequisites:

```
cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

# Setup required sysctl params, these persist across reboots.
cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables  = 1
net.ipv4.ip_forward                 = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF

# Apply sysctl params without reboot
sudo sysctl --system
```

Ref: https://kubernetes.io/docs/setup/production-environment/container-runtimes/#containerd

kubernetes by Google

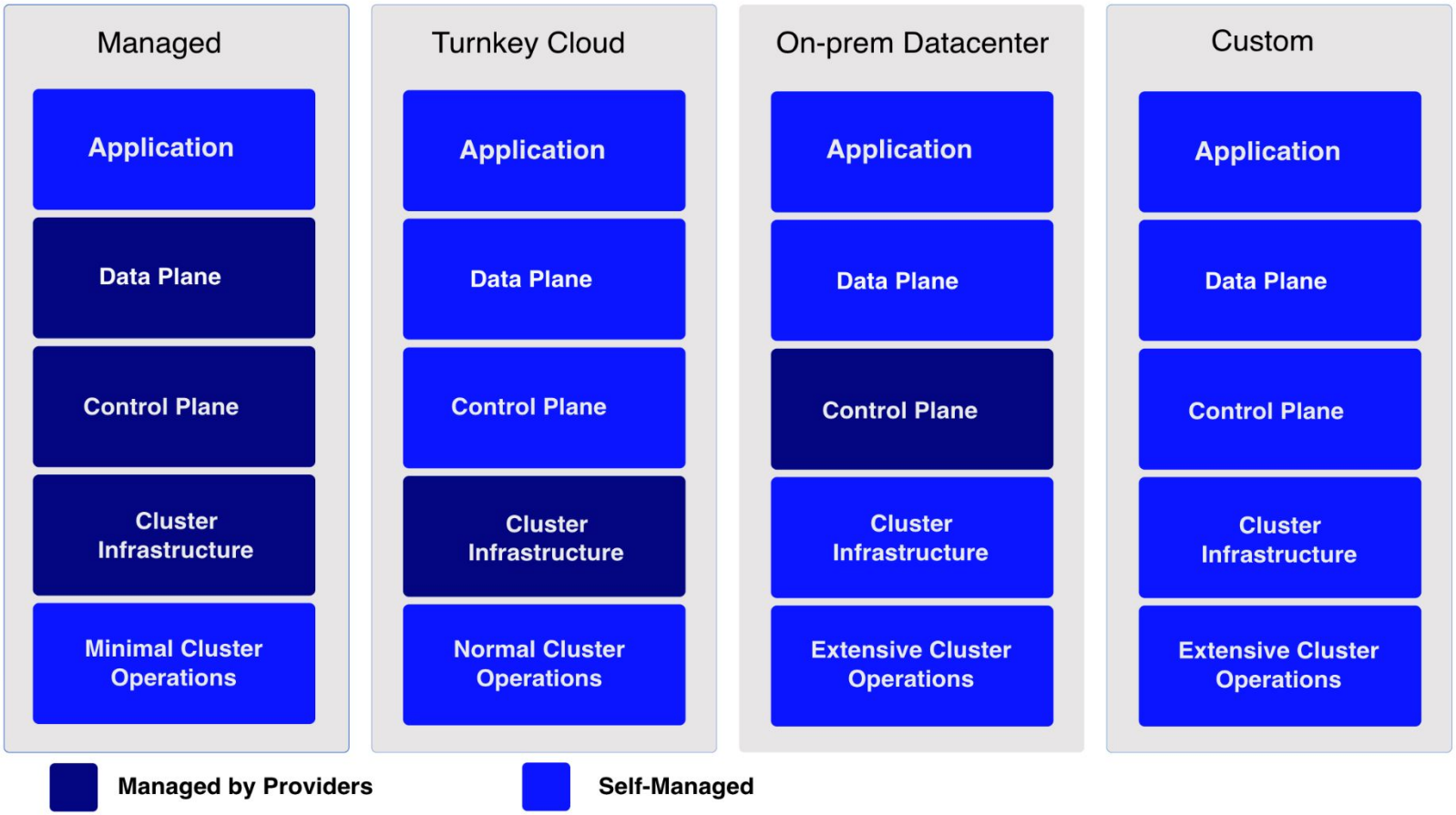
Kubernetes Environment

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

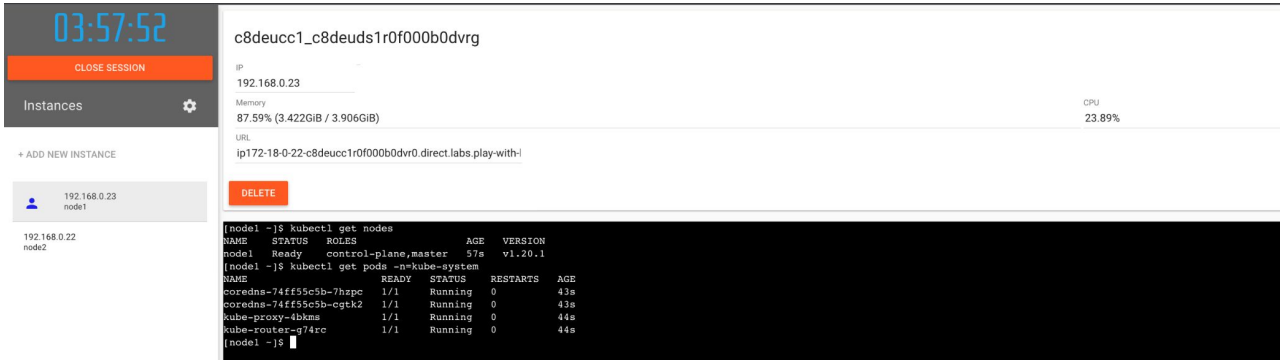
Kubernetes: Production Workload Orchestration

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

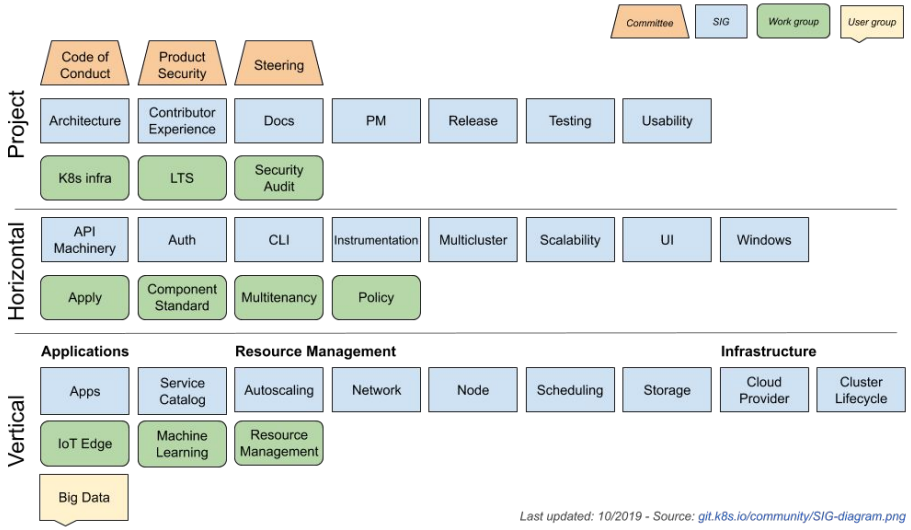
Introduction to Kubernetes

Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes

Containers, that'll fix it.

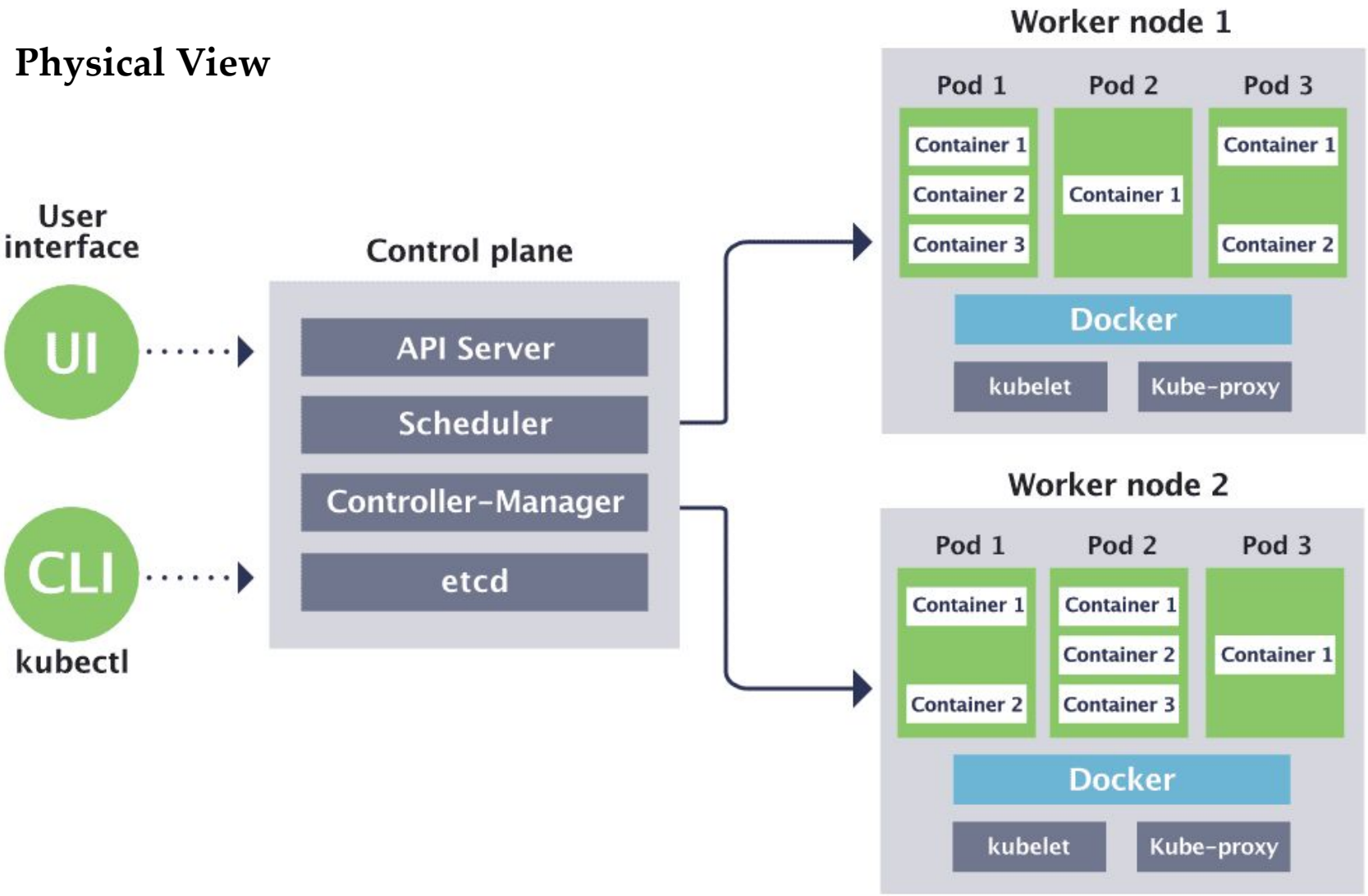


Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes

Physical View

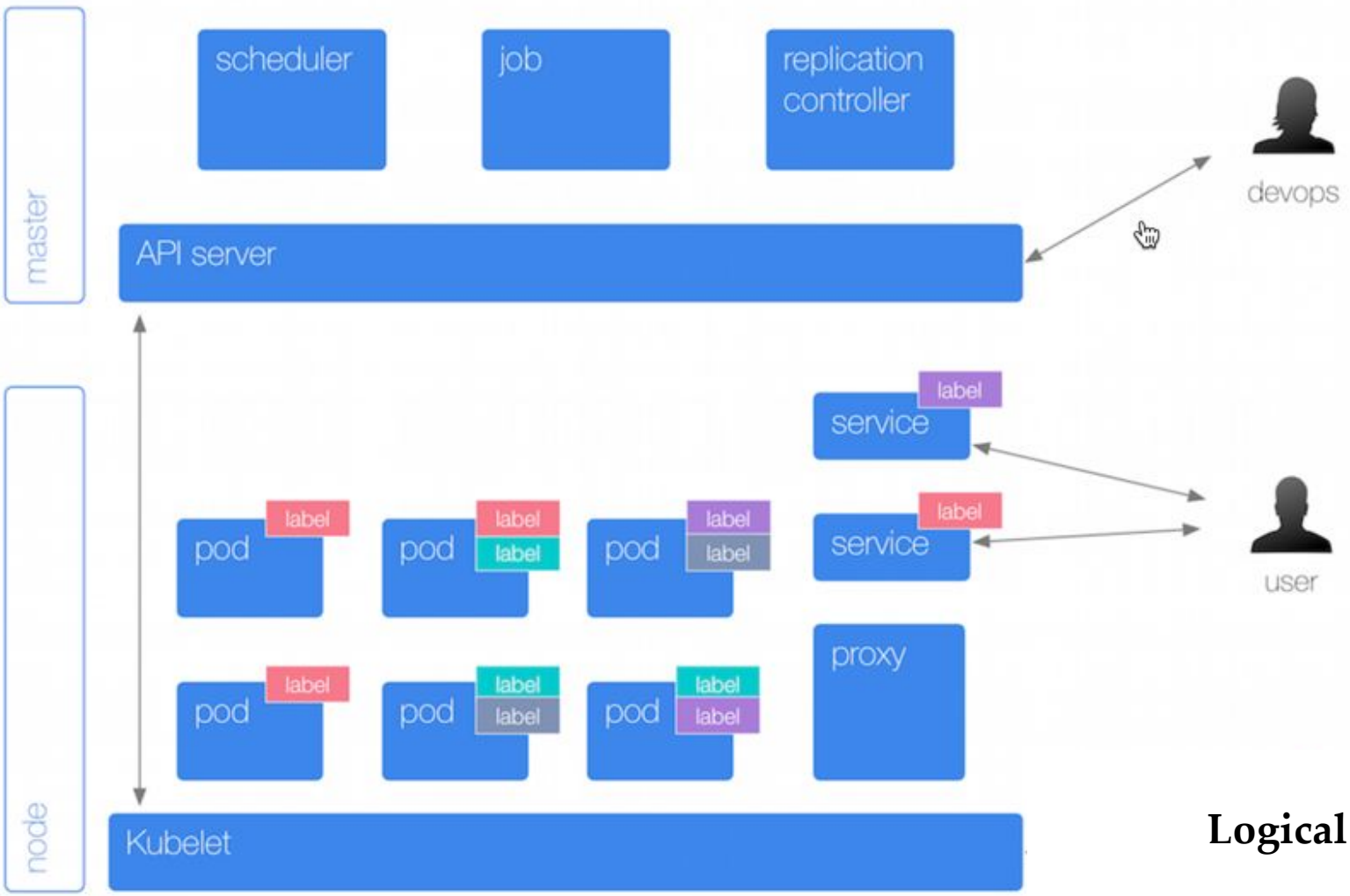


Ref: https://www.cncf.io/blog/2019/08/19/how-kubernetes-works/

Kubernetes: Production Workload Orchestration

kubernetes by Google

Introduction to Kubernetes



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

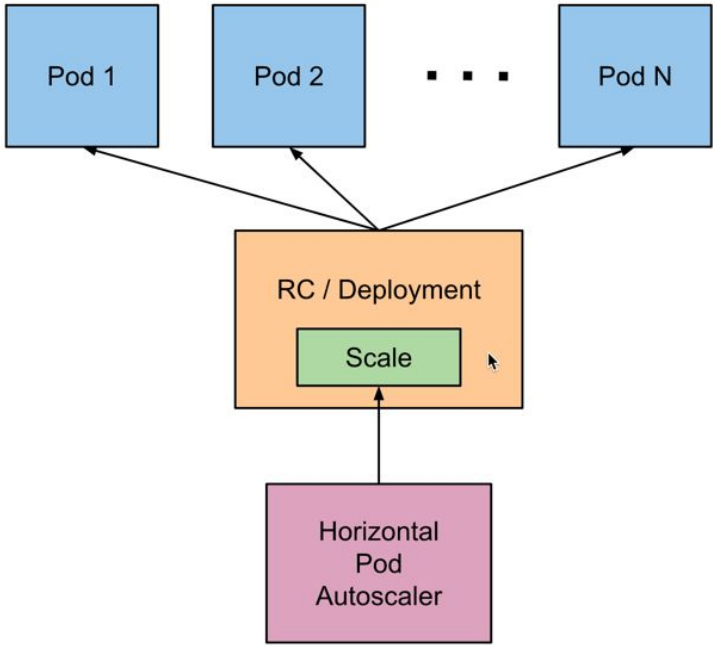
Introduction to Kubernetes

- 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>
```

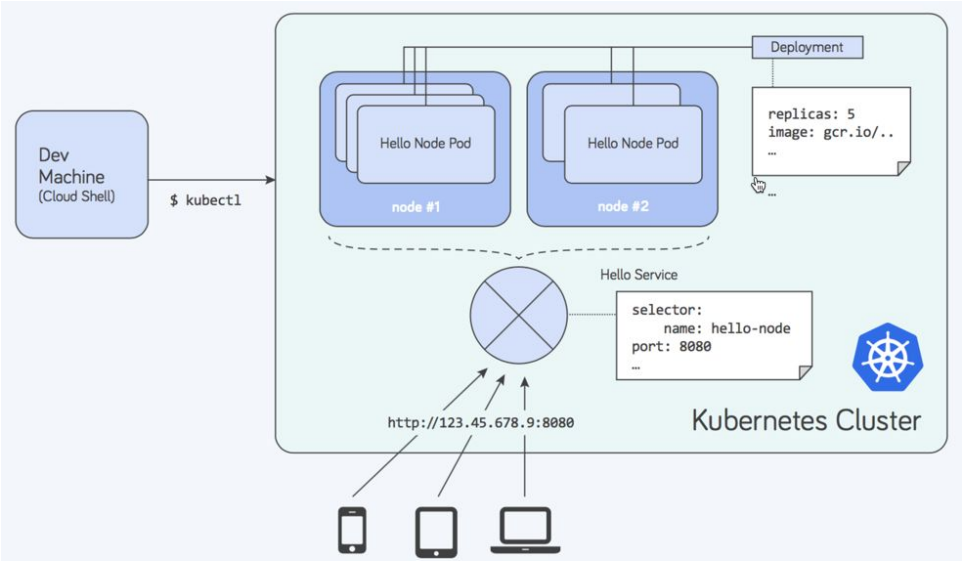
Introduction to Kubernetes

- 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)



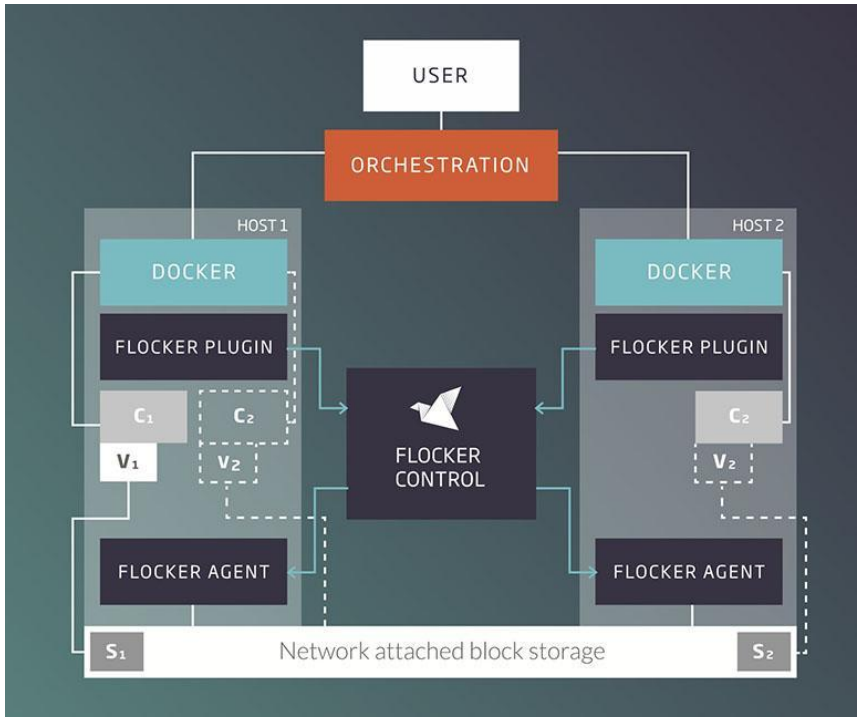
Introduction to Kubernetes

- 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, Azue Disk etc)

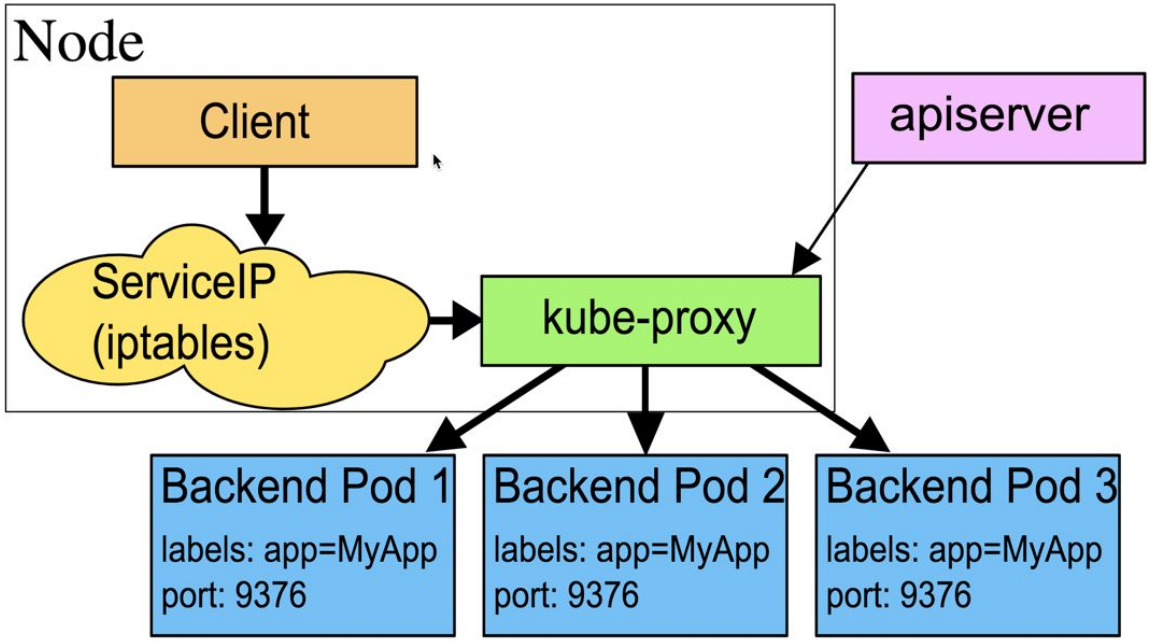


Introduction to Kubernetes

- 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)



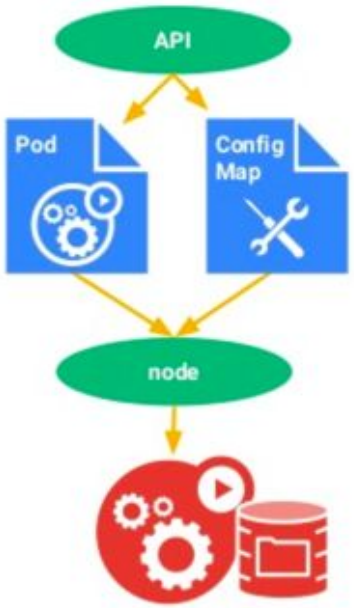
Introduction to Kubernetes

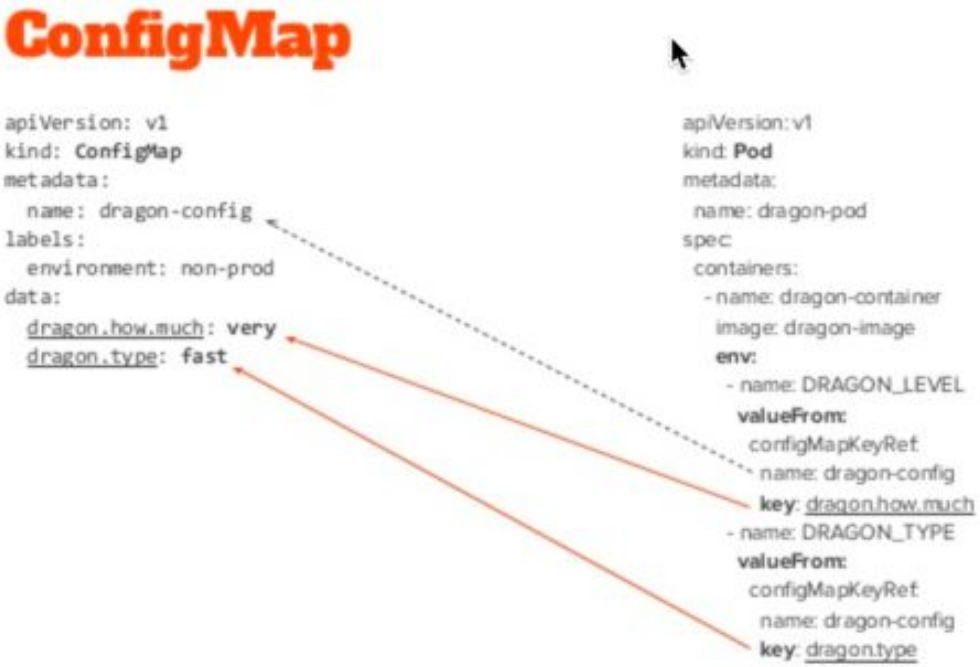
- Secret and Configuration Management
  - Kubernetes can keep confidential data (such as username/password) for running application on encrypt format.
  - Can reference on Pods instead plain text configuration.

```
$ 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



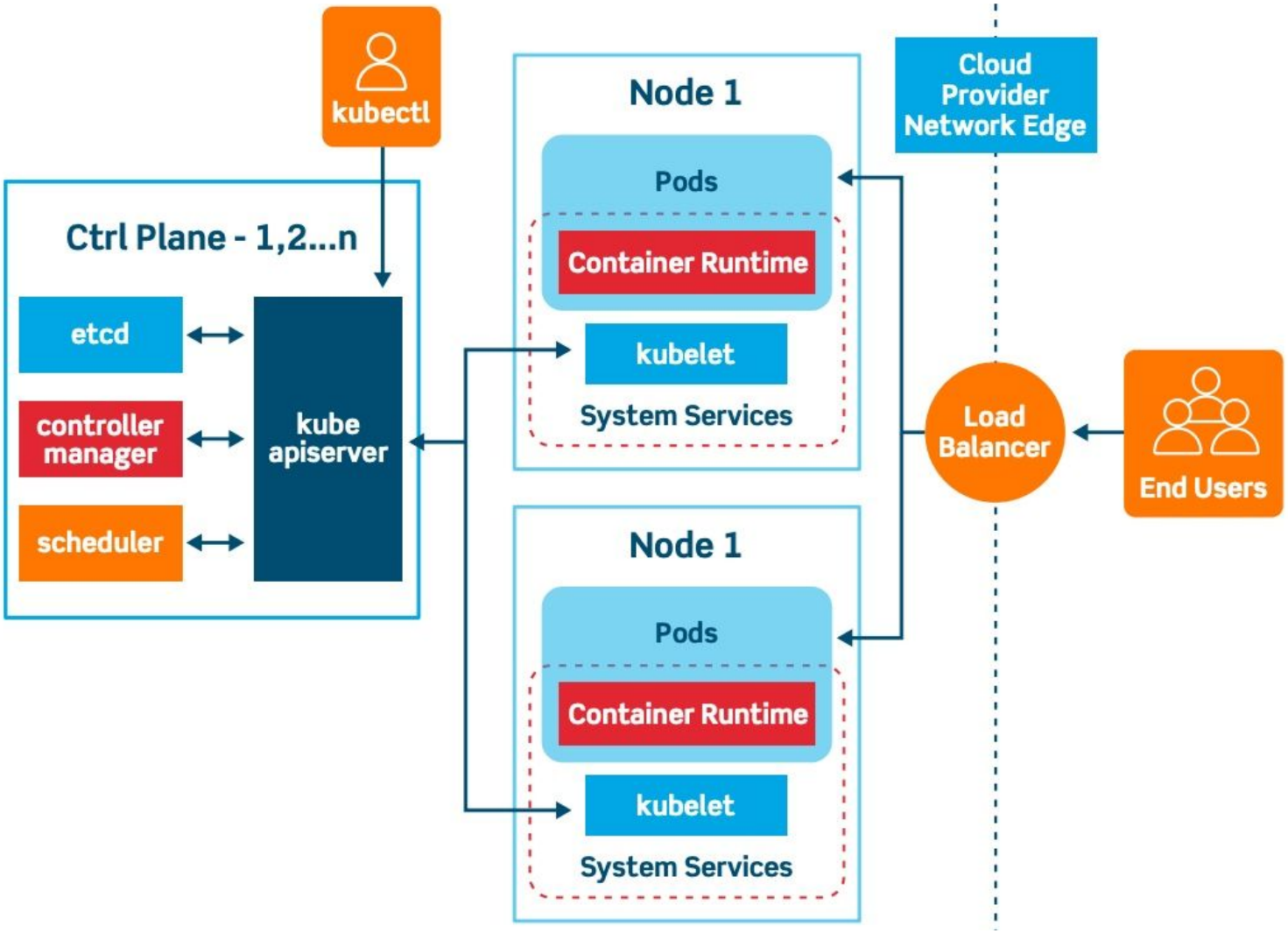


Introduction to Kubernetes

- 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

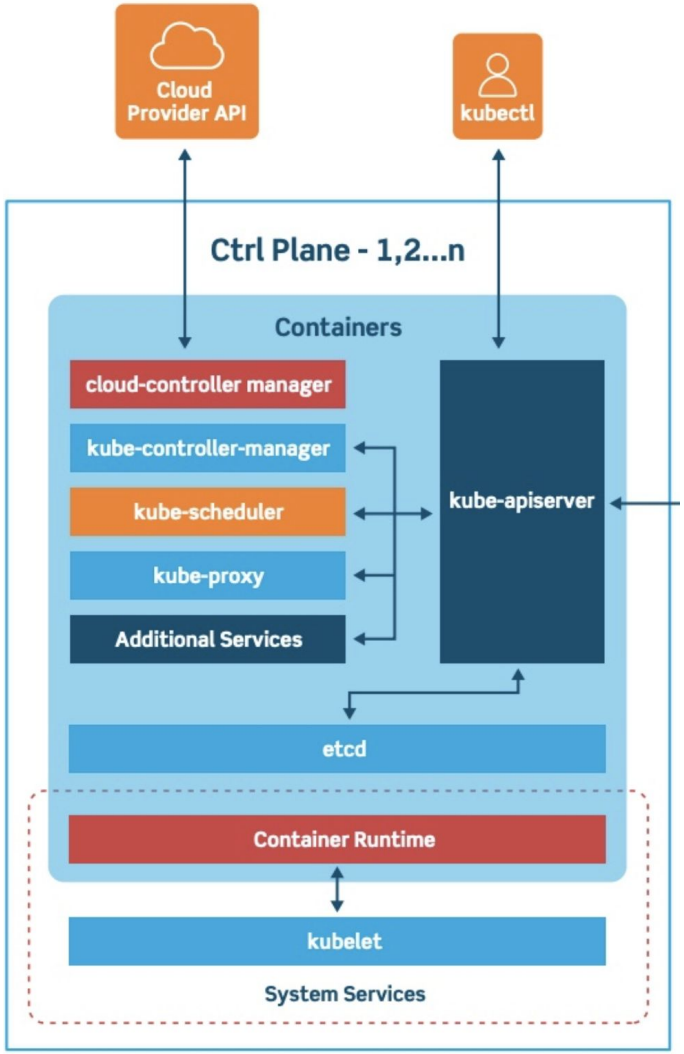
System Architecture

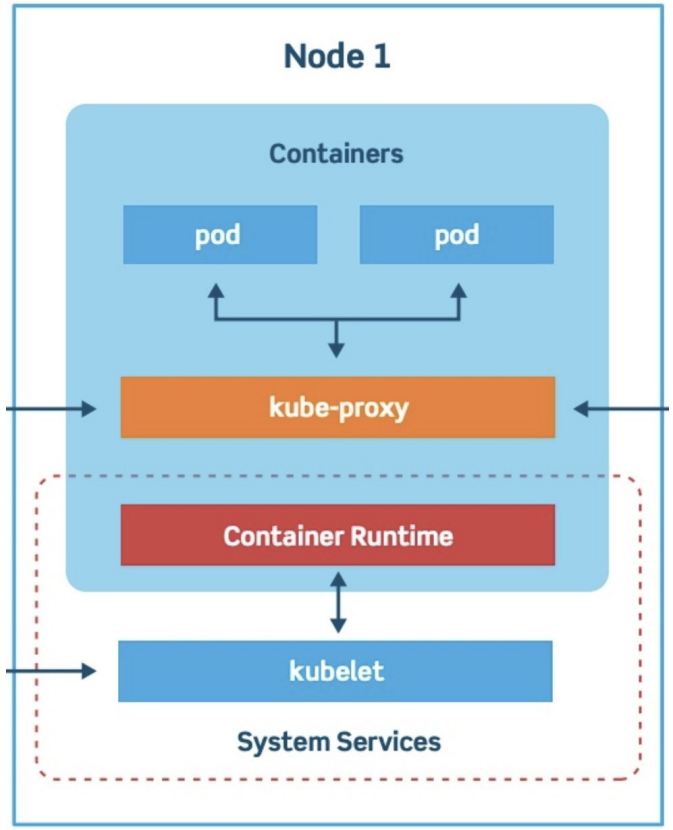
System Architecture



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/

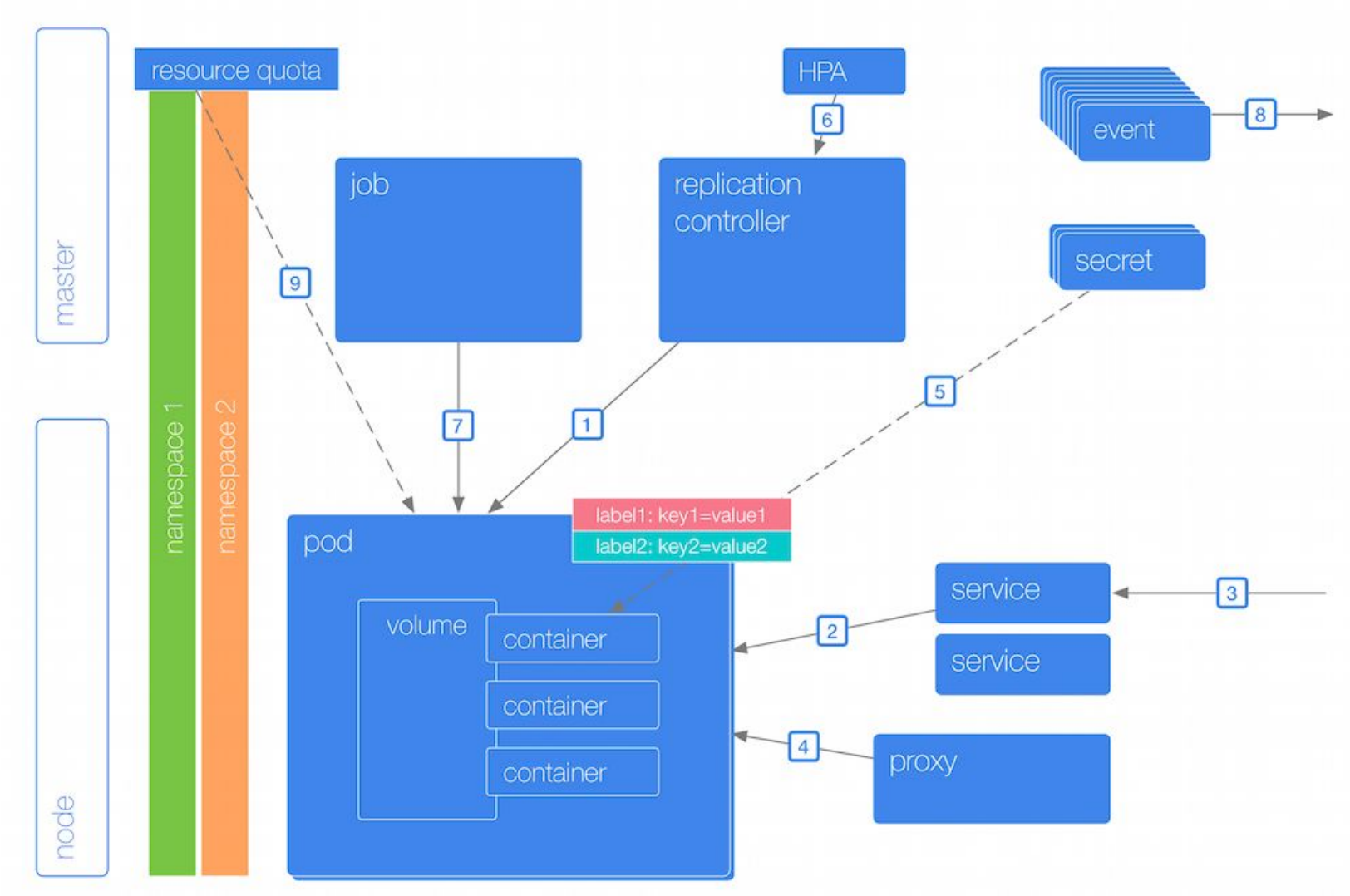
System Architecture

- 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

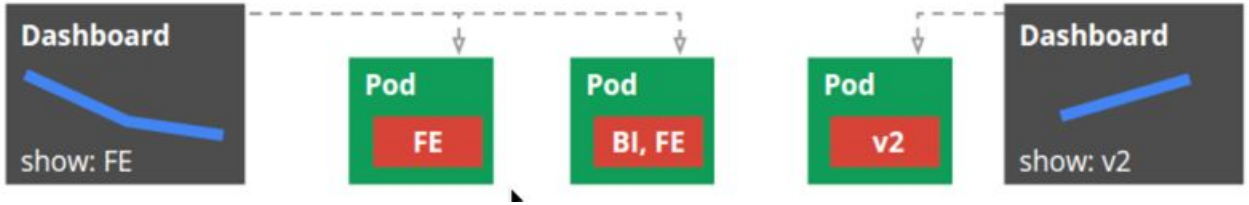
- Kubernetes Node
  - Pods: Pods is smallest deployable units of computing that can be created and manage in kubernetes (Pods != Container).
  - Container runeimt: Container engine for run container. Kubernetes support the runtime with standard CRI. such as cri-o, containerd
  - Kubelet: Agent on node 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"
```

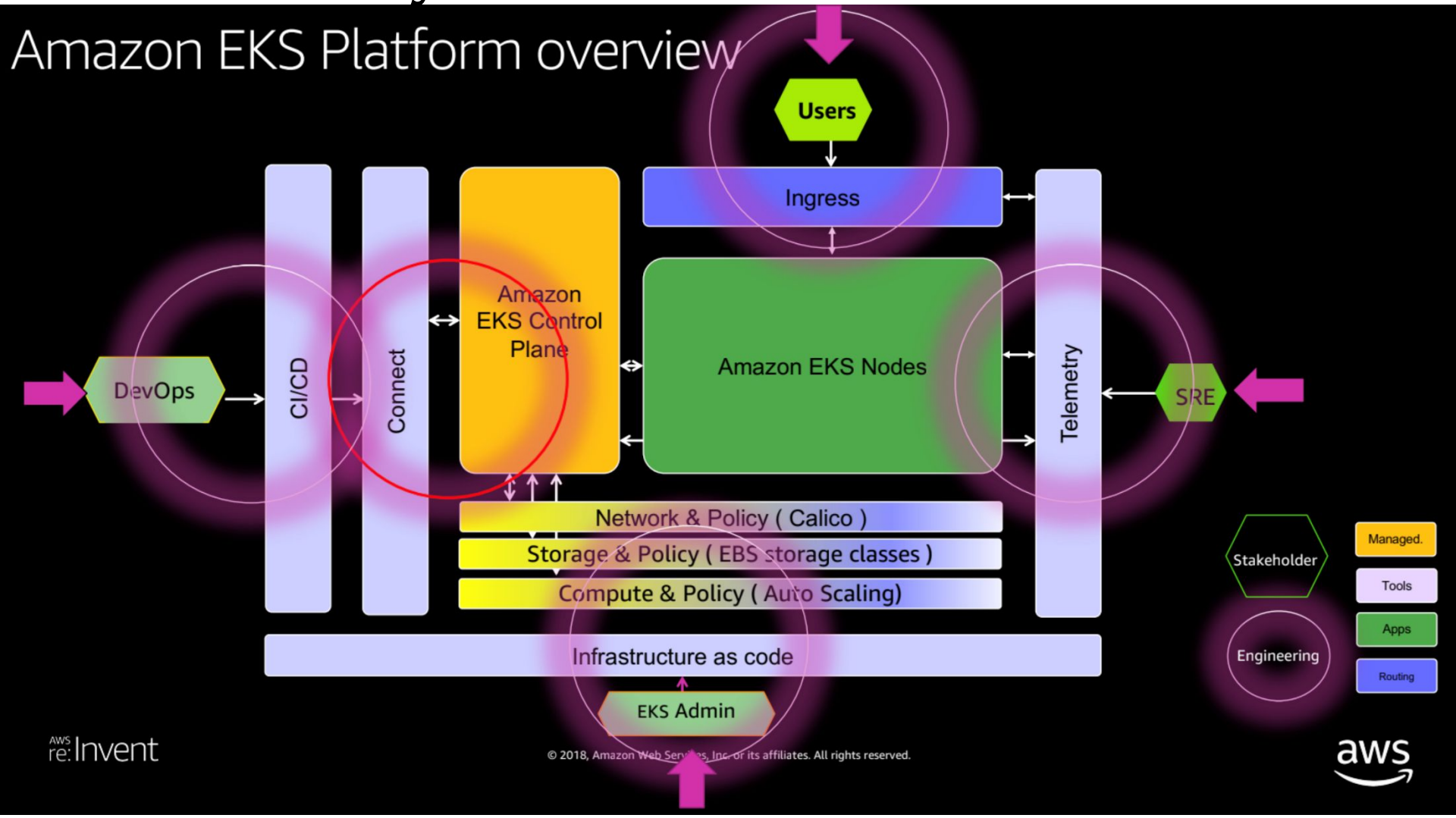
```
"partition" : "customerA", "partition" : "customerB"
```

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

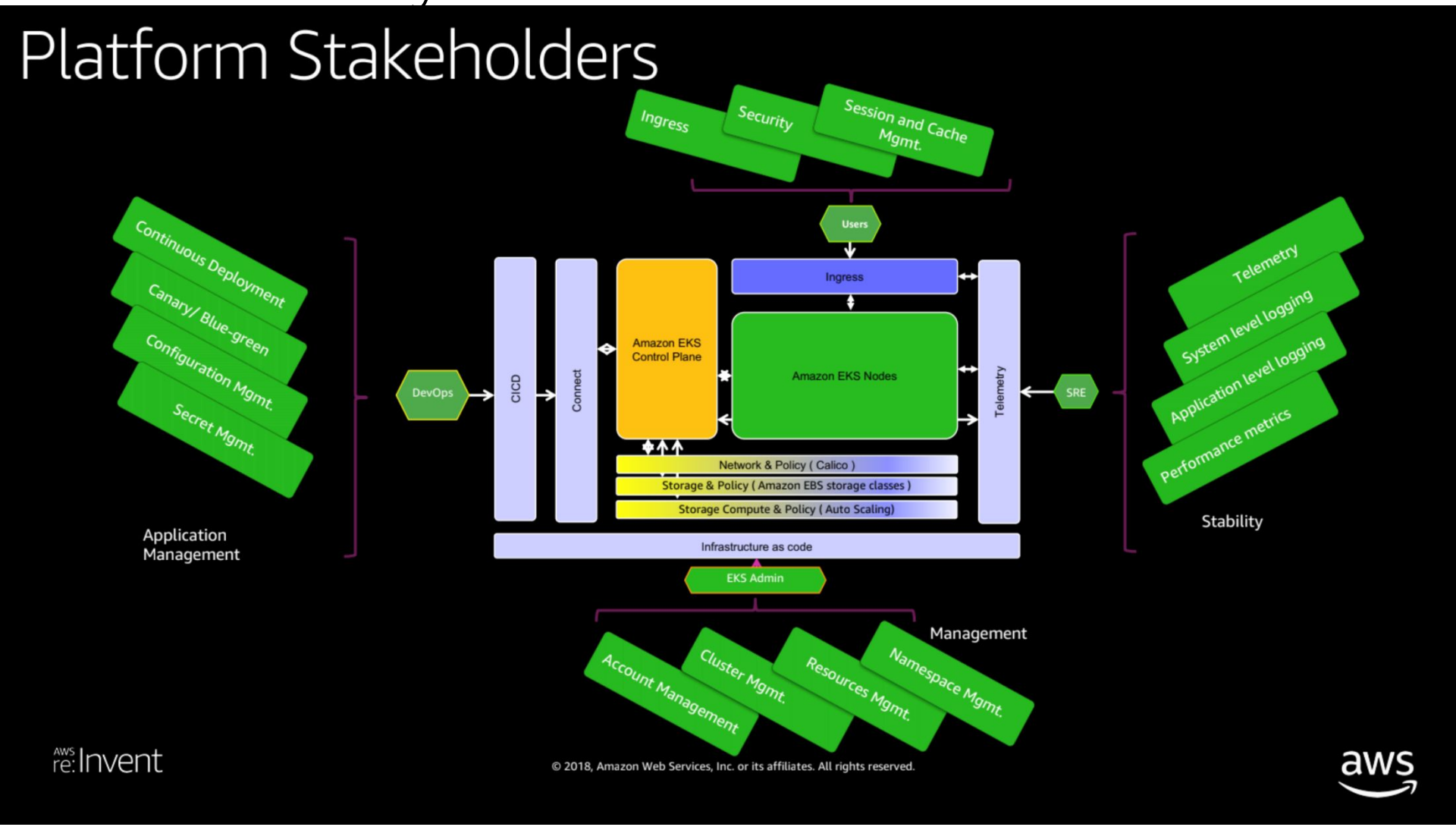
System Architecture

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

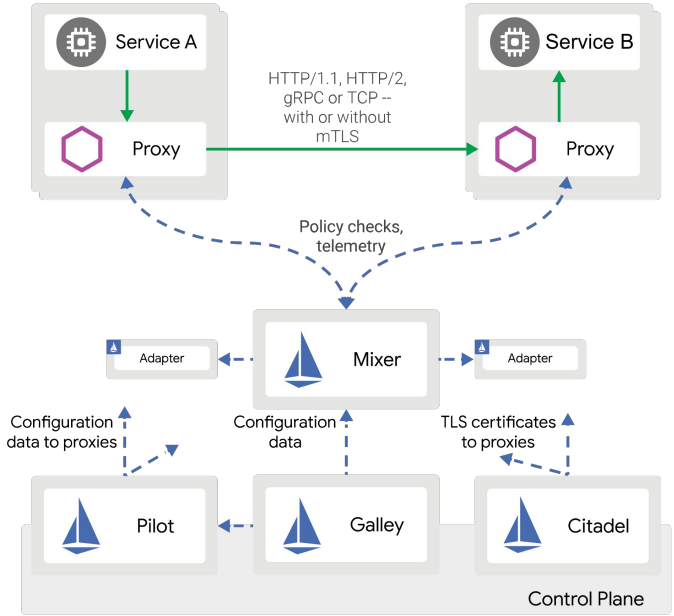
Pods, Container and Services

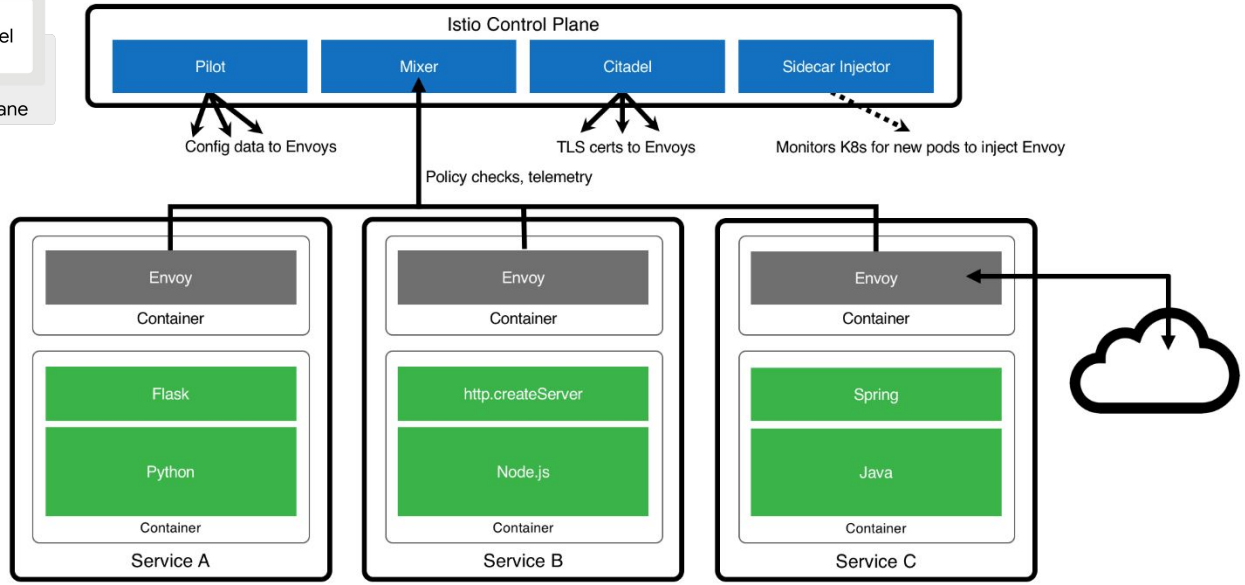
- Pods vs Container
  - All container on same Pods will share:
    - Process ID (PID)
    - Network access (Communicate to each other via "localhost")
    - Internal Process Command (IPC)
    - Unix Time-Sharing (UTS)
    - Hostname
    - IP Address/Ports

  - 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



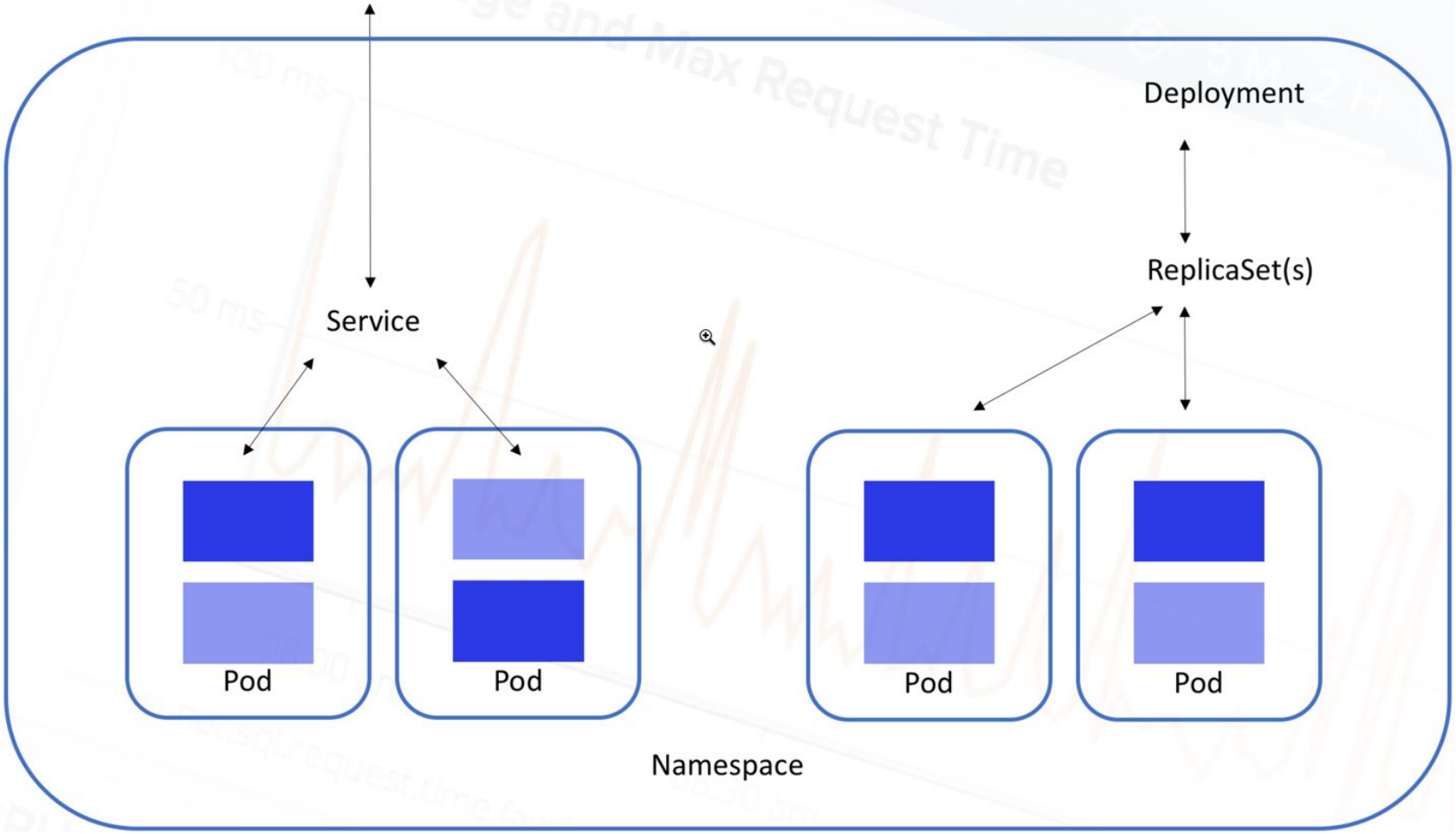


Kubernetes: Production Workload Orchestration

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 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$
```

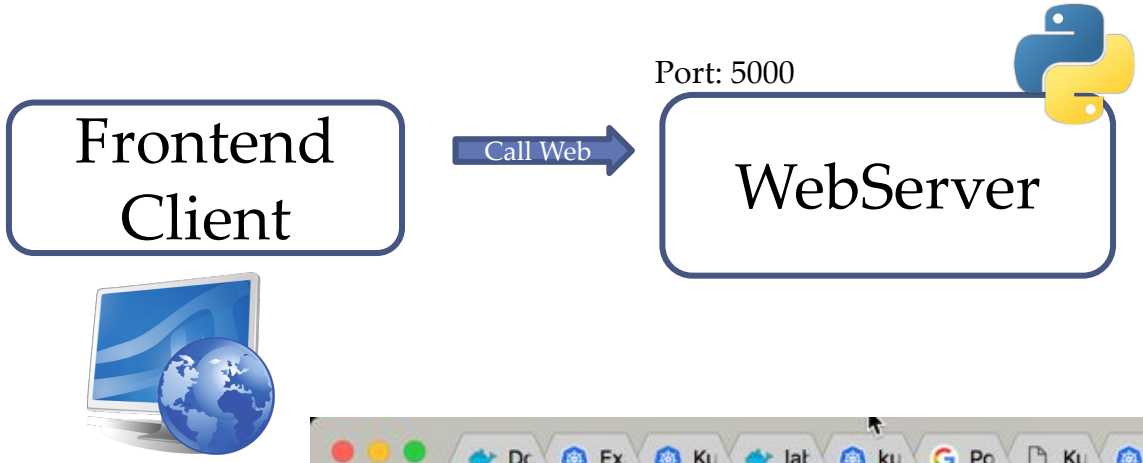
- Remove Pod/Service

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
```

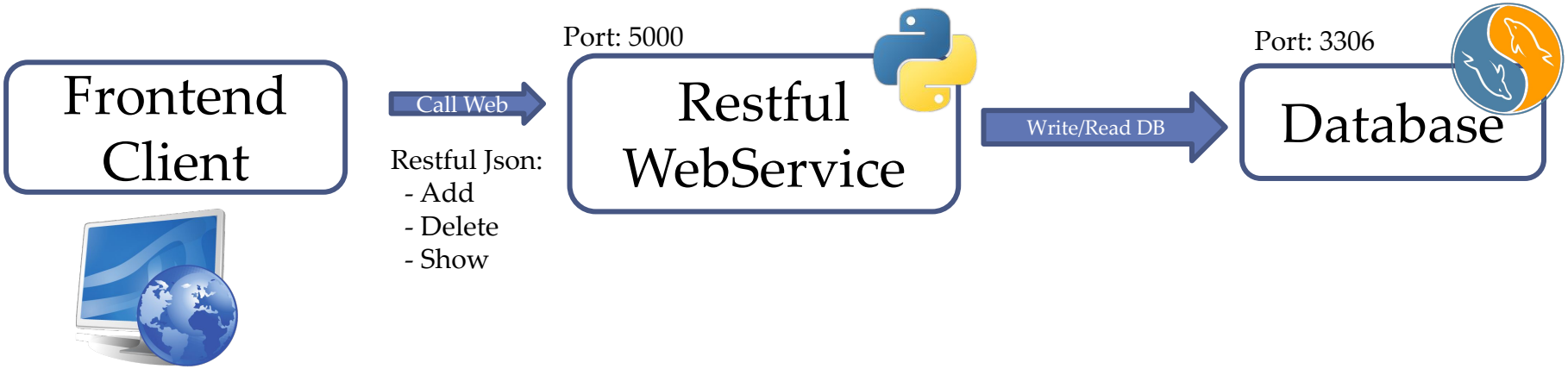
Workshop: Pods & Service

- Part 1: Deploy simple web pods with single container



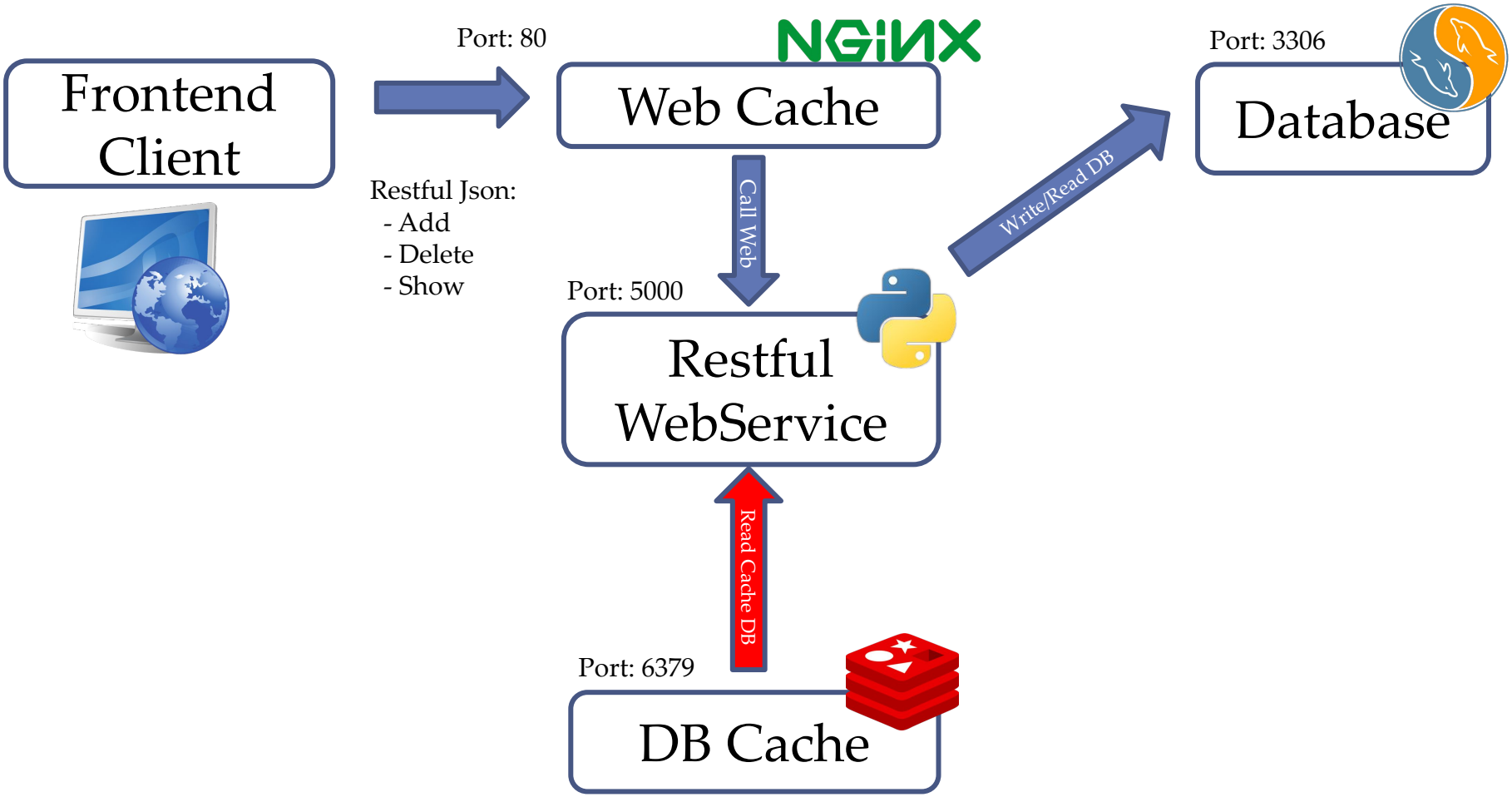
Pods, Container and Service

- 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

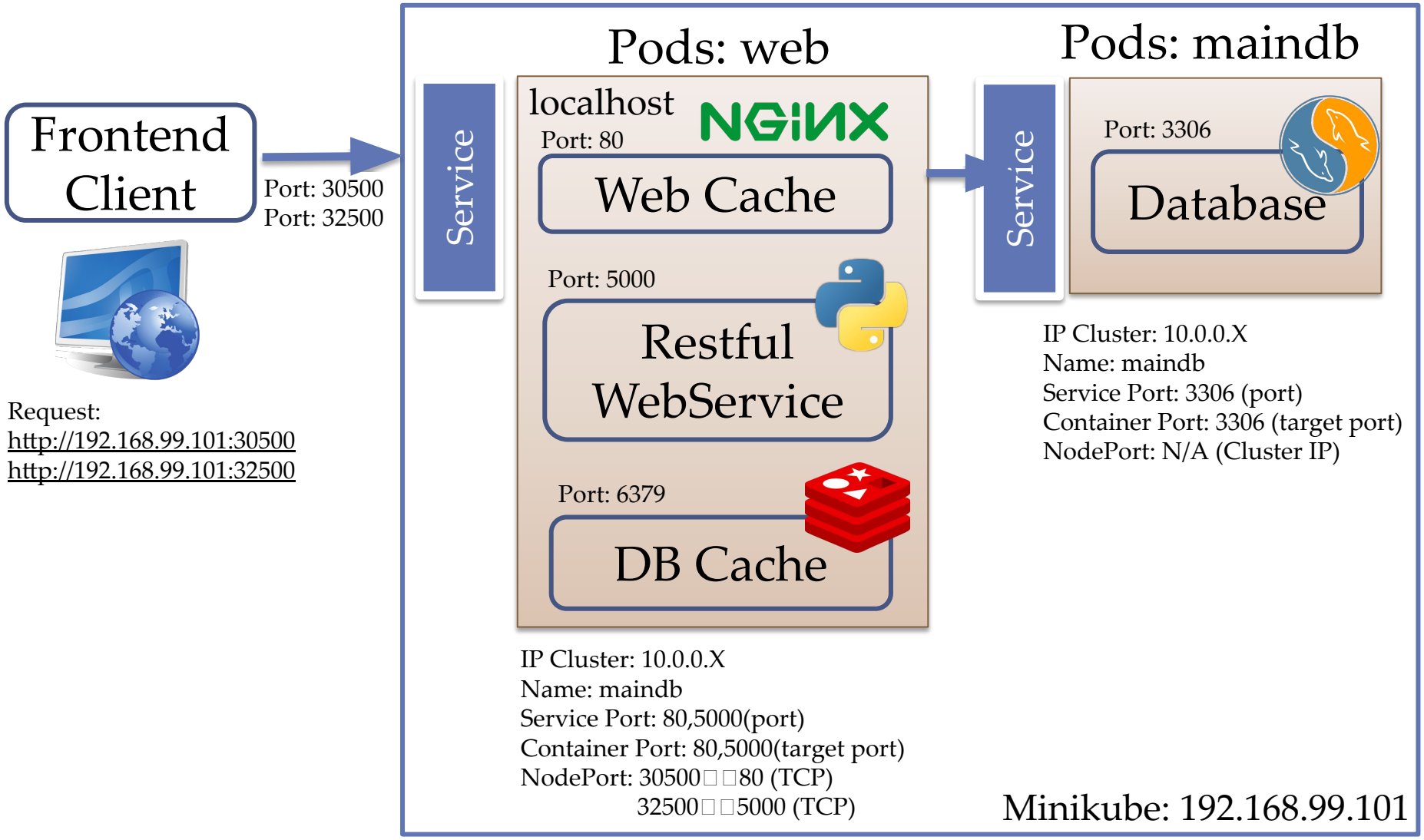
- Database and Database Cache

```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

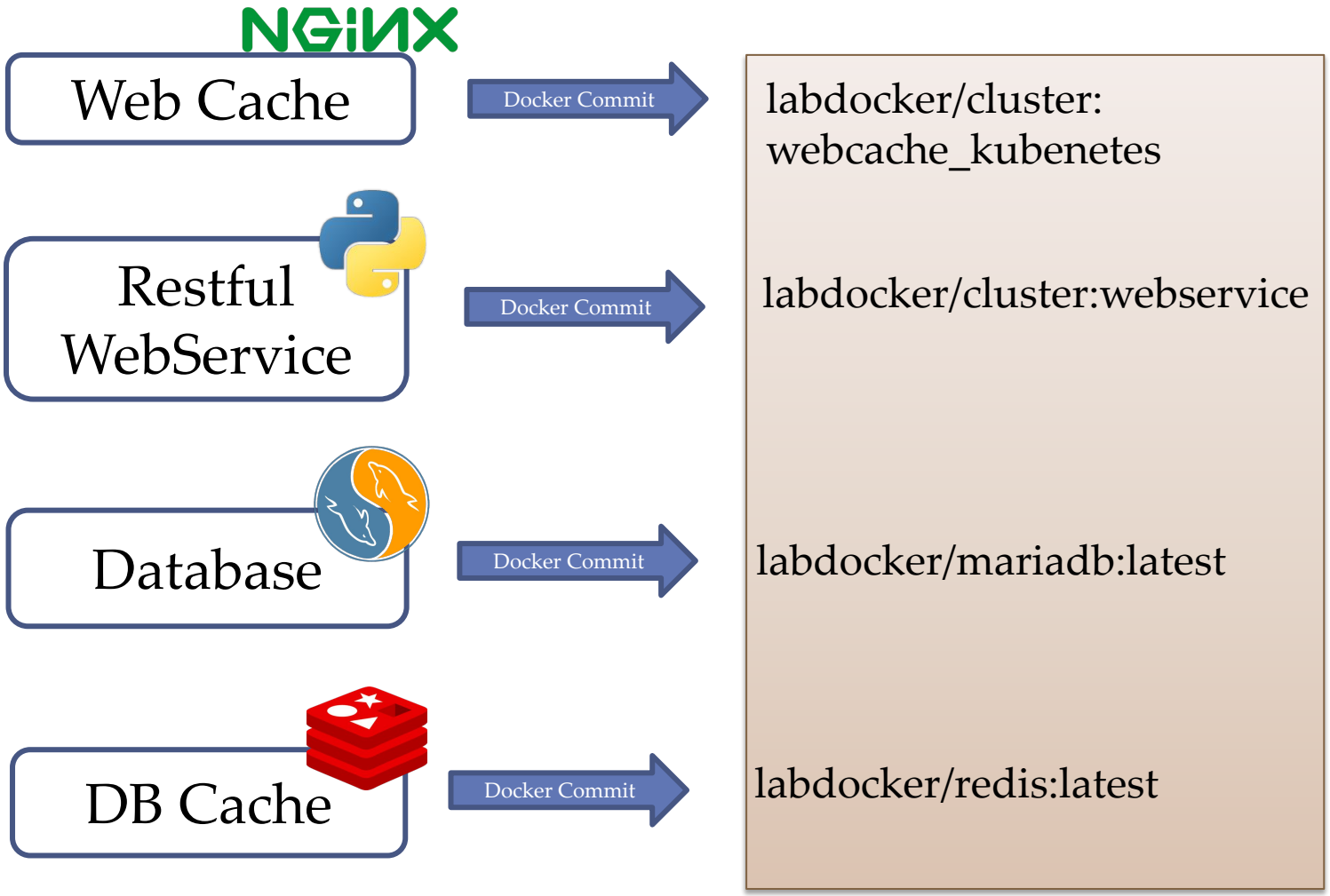
Pods, Container and Service



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)

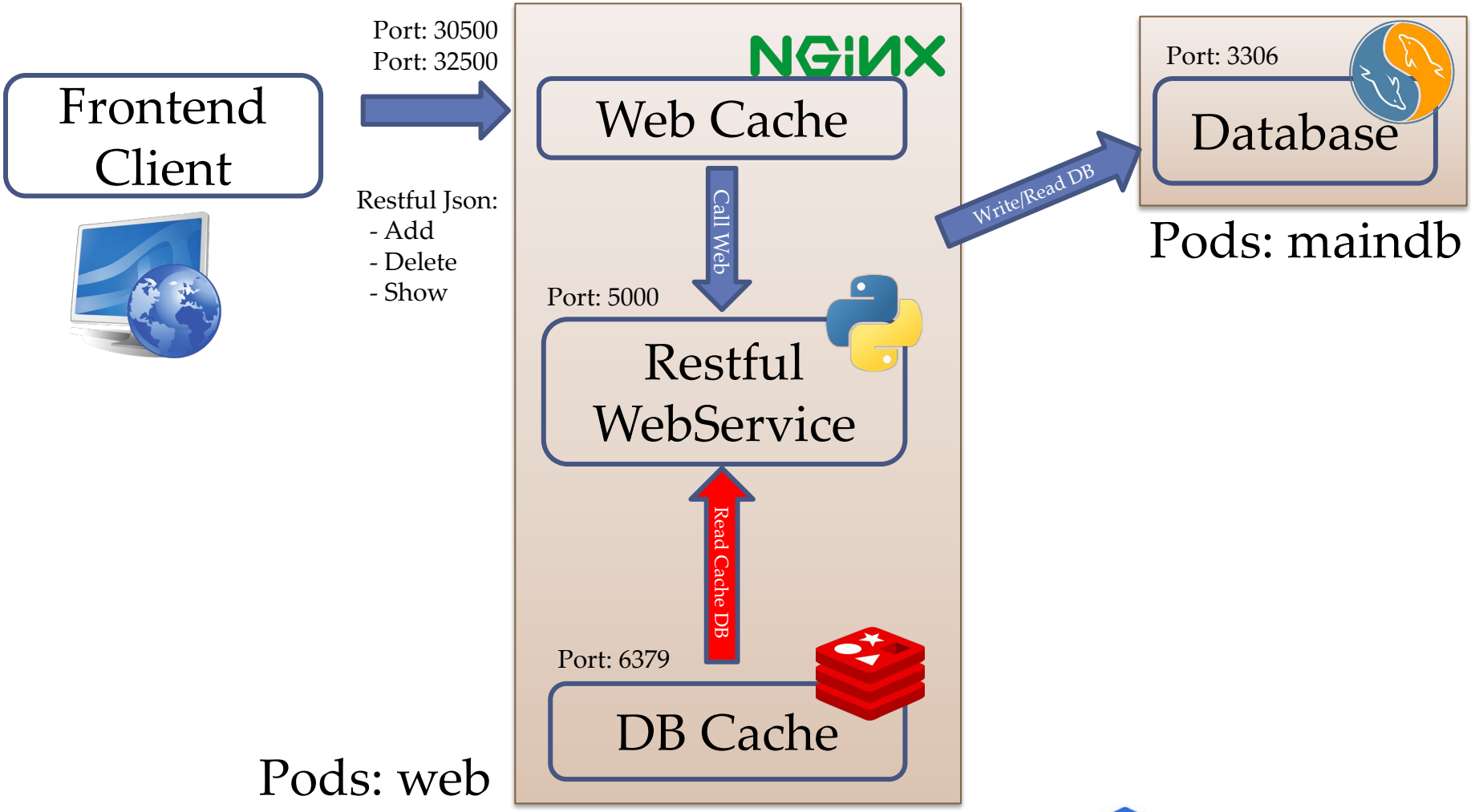
```
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/removeuser/1
########### Record was deleted (Both Database Cache) ###########
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port/users/removeuser/2
########### Record was deleted ###########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

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
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

METADATA APIS

ControllerRevision v1 apps
CustomResourceDefinition v1beta1 apiextensions.k8s.
Event v1 core
LimitRange v1 core
HorizontalPodAutoscaler v1 autoscaling

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/v1.15/#-strong-api-overview-strong-

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
```

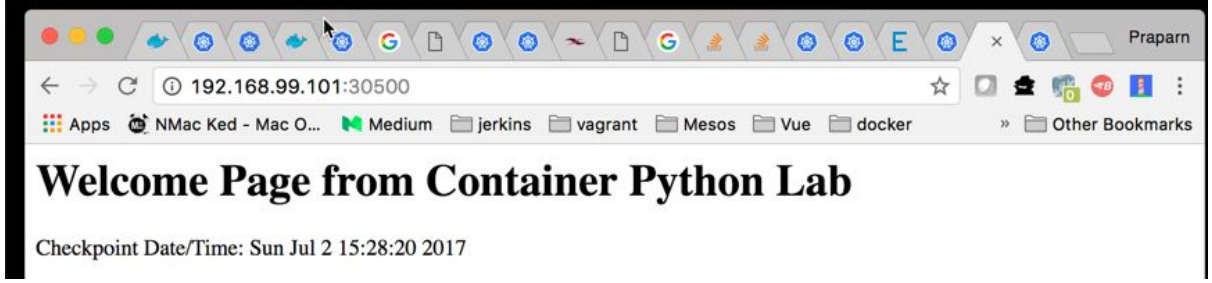
Service “web”

```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$
```

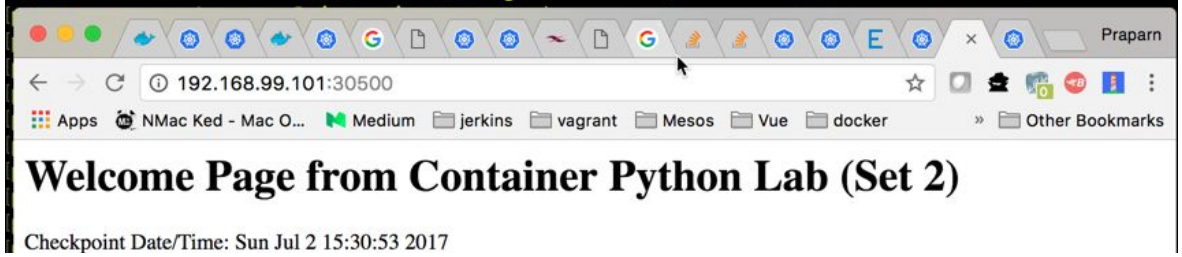



```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)                    AGE
kubernetes   10.0.0.1      <none>        443/TCP                    5d
maindb       10.0.0.236    <none>        3306/TCP                   1m
web          10.0.0.212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

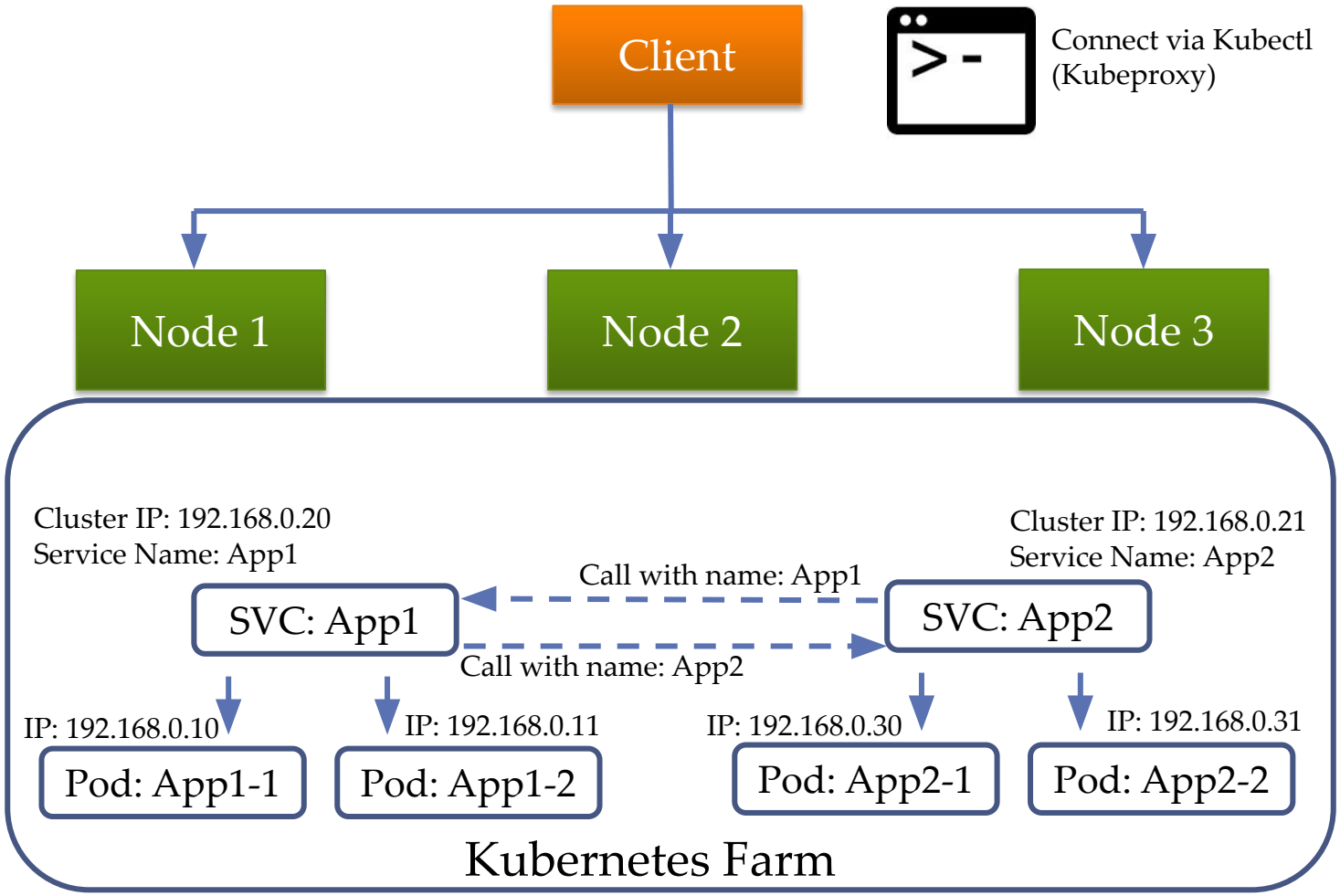
- Services:
  - Publish Service Type:
    - CLUSTER-IP (Default): blind port with ip address of cluster (Accessible from internal cluster system)
    - NodePort: blind port with ip address of node. By default kubernetes will random port (30000-32757). If we need to specify set the option: "nodePort: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

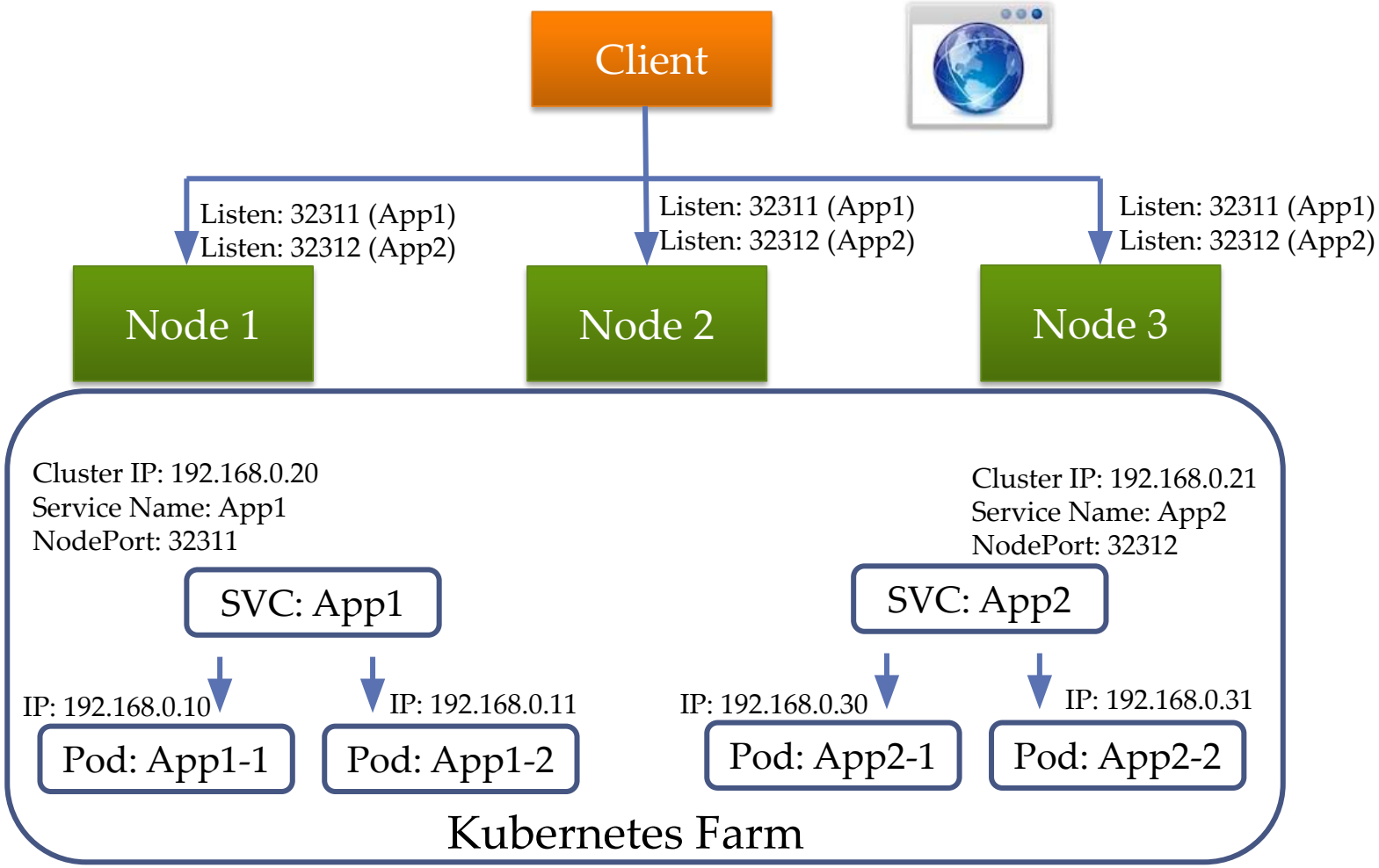
Pods, Container and Services

- Services: ClusterIP



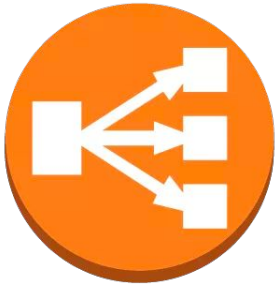
Pods, Container and Services

- Services: NodePort



Pods, Container and Services

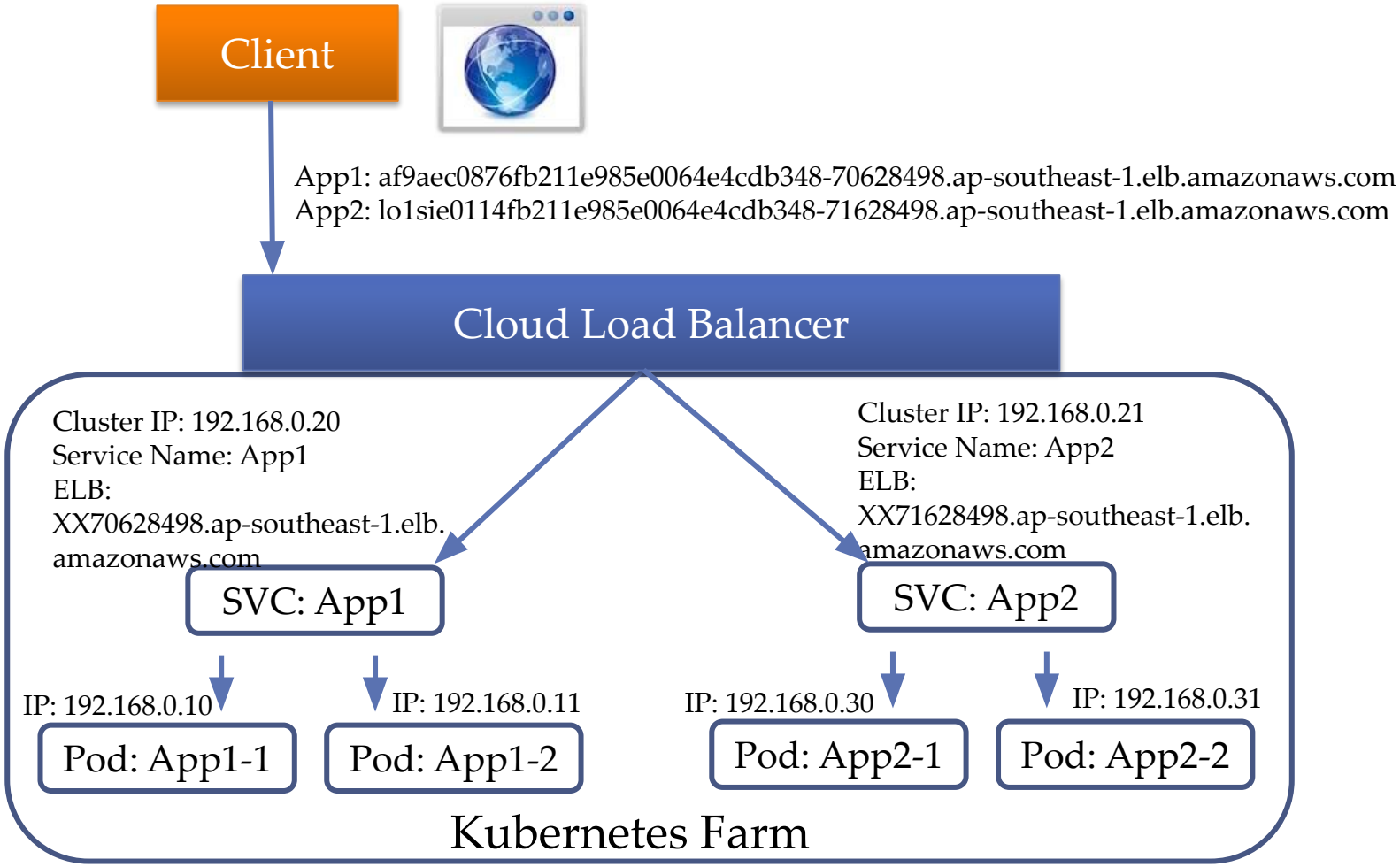
- Services
  - Publish Service Type:
    - LoadBalance:
      - Use load balancer from external cloud provider for intercept traffic
      - Manage by Kubernetes Itself
      - Flexible for use facilities on cloud



```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



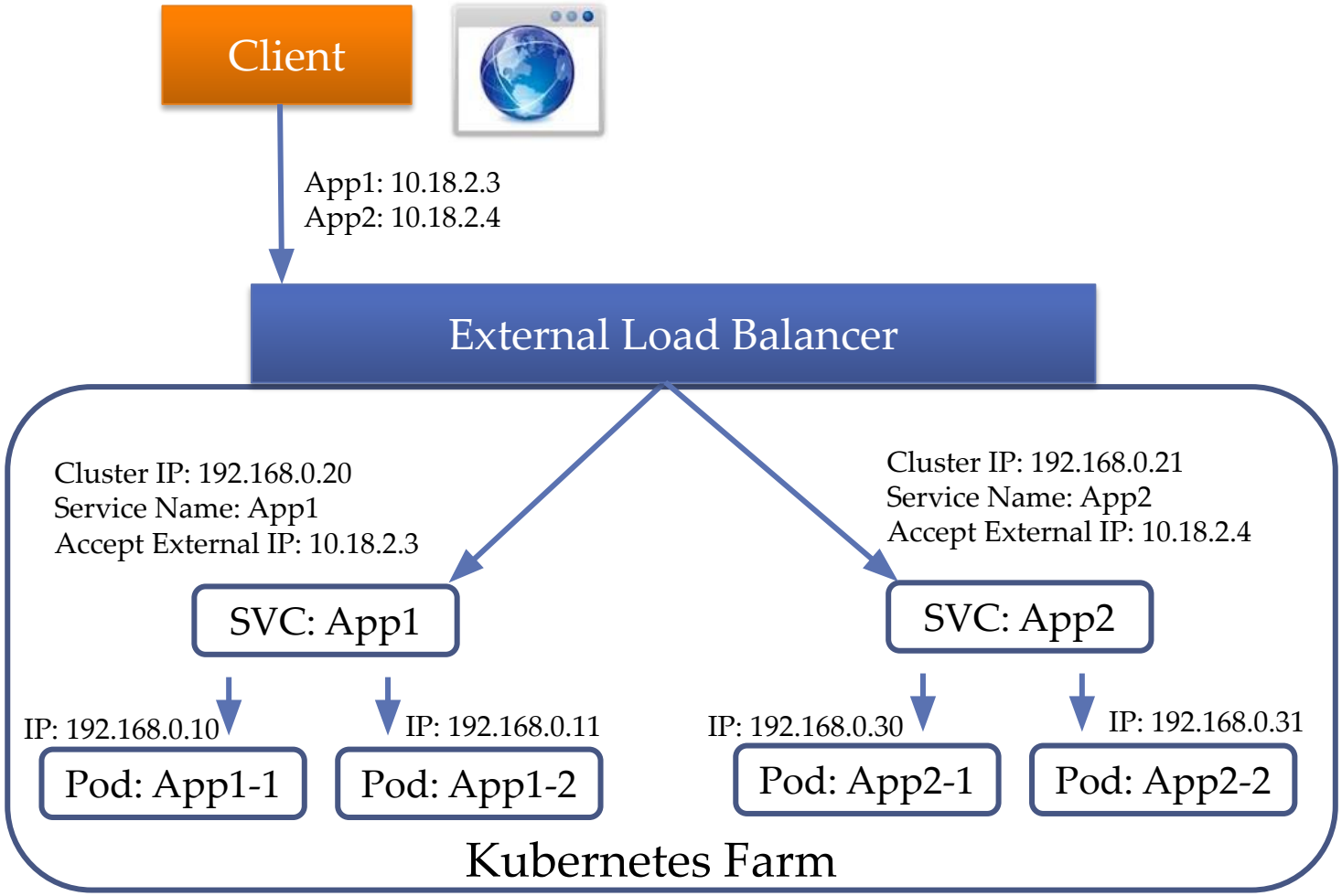
Pods, Container and Services

- Services:
  - 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

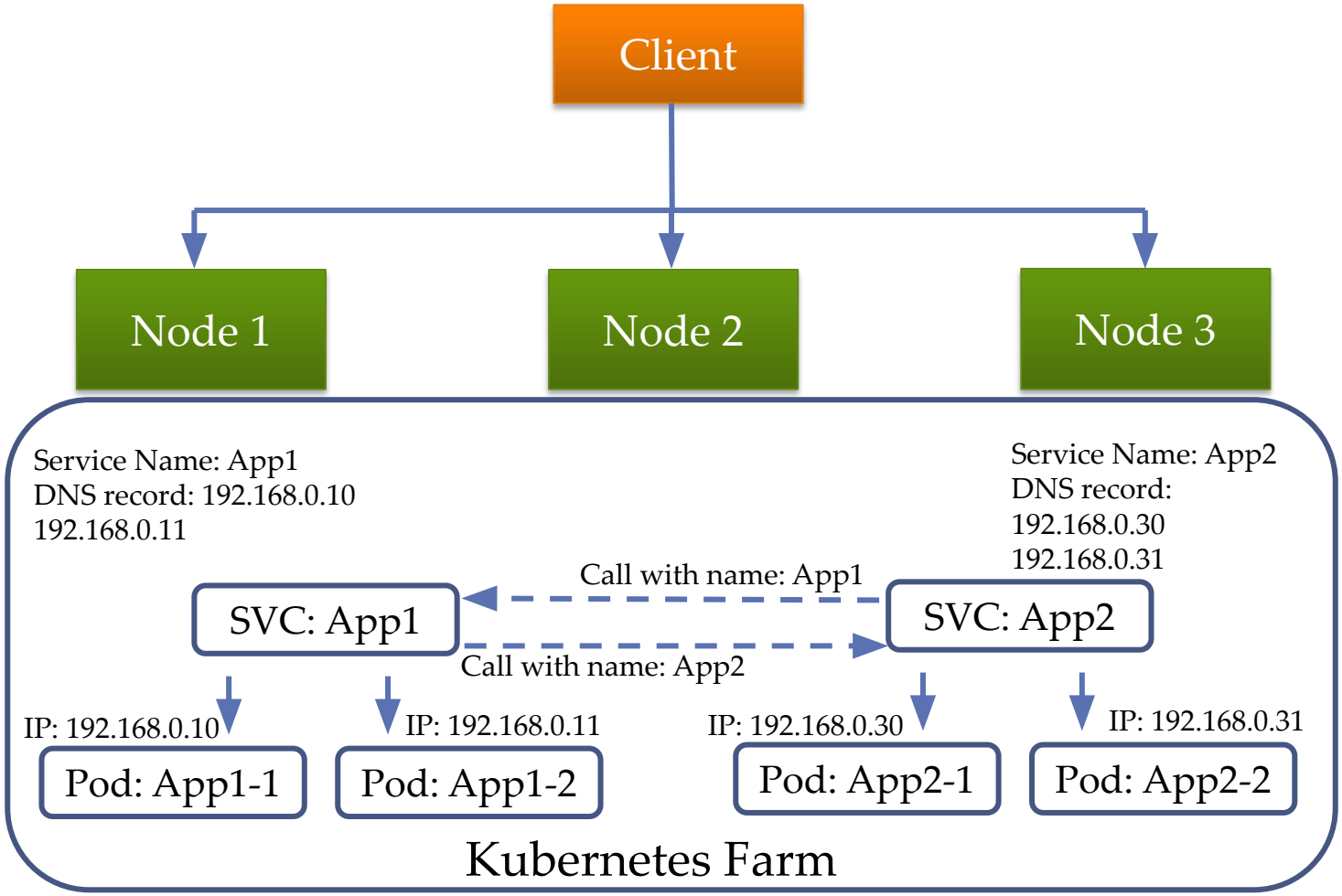
- Services
  - Publish Service Type:
    - Headless Service (StatefulSet): service will just dns round-robin for all statefulset pods's ip address

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)

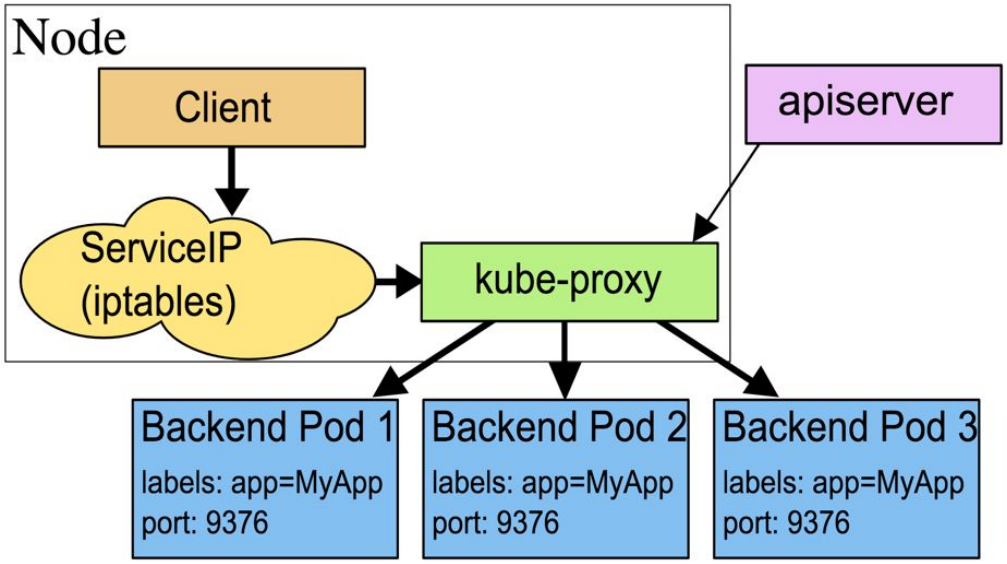
```yaml
kind: Service
apiVersion: v1
metadata:
  name: my-service
  namespace: prod
spec:
  type: ExternalName
  externalName: my.database.example.com
```

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

Pods, Container and Services

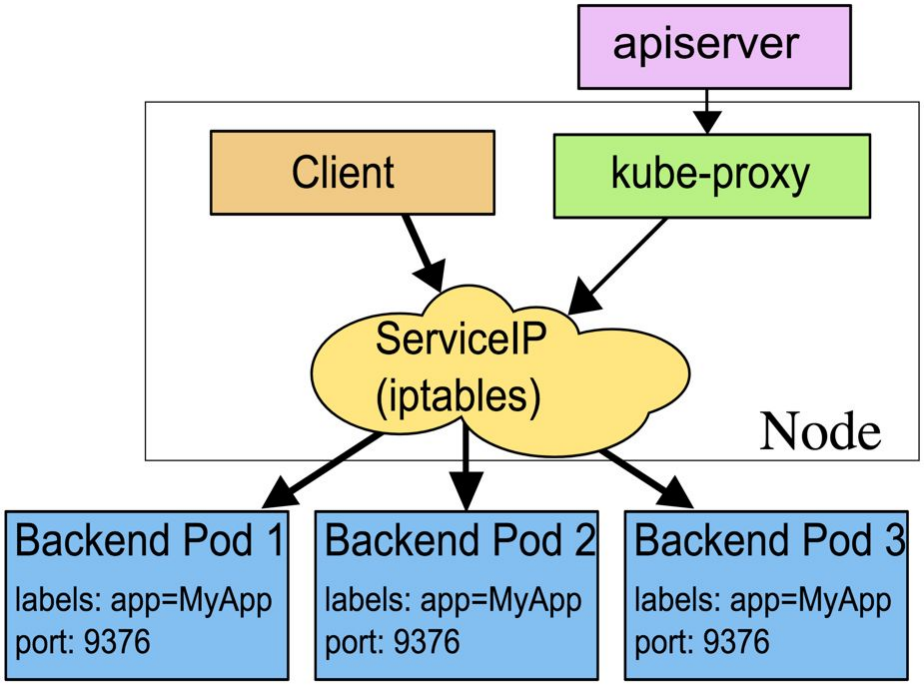
- kube-proxy (proxy-mode)
  - Every node on K8S will use kube-proxy for response to service
    - Initial Virtual IP for service (except External Name)
    - Vary "proxy-mode" for support in kube-proxy

  - 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



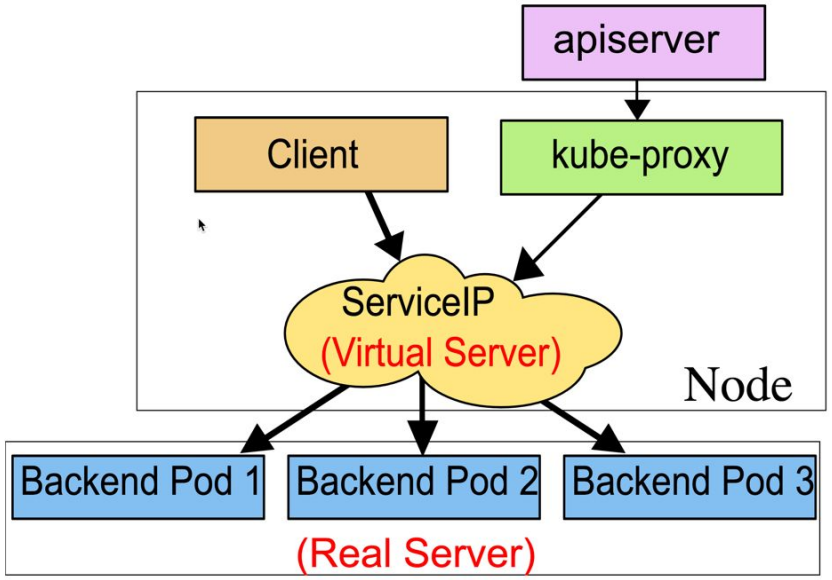
Pods, Container and Services

- 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)
    - Not support in case pods fail



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)

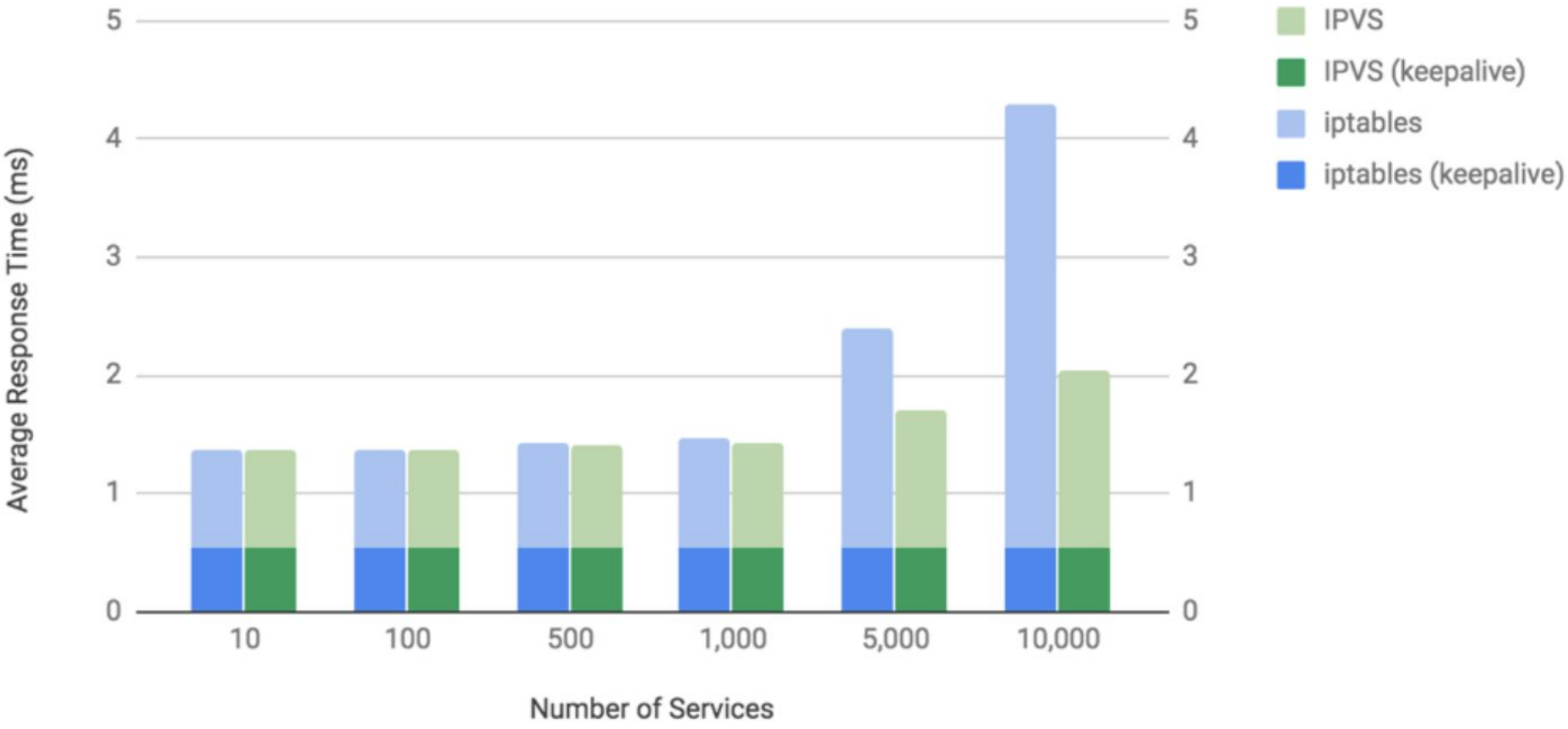
```
ip_vs
ip_vs_rr
ip_vs_wrr
ip_vs_sh
nf_conntrack_ipv4
```

```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

- 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

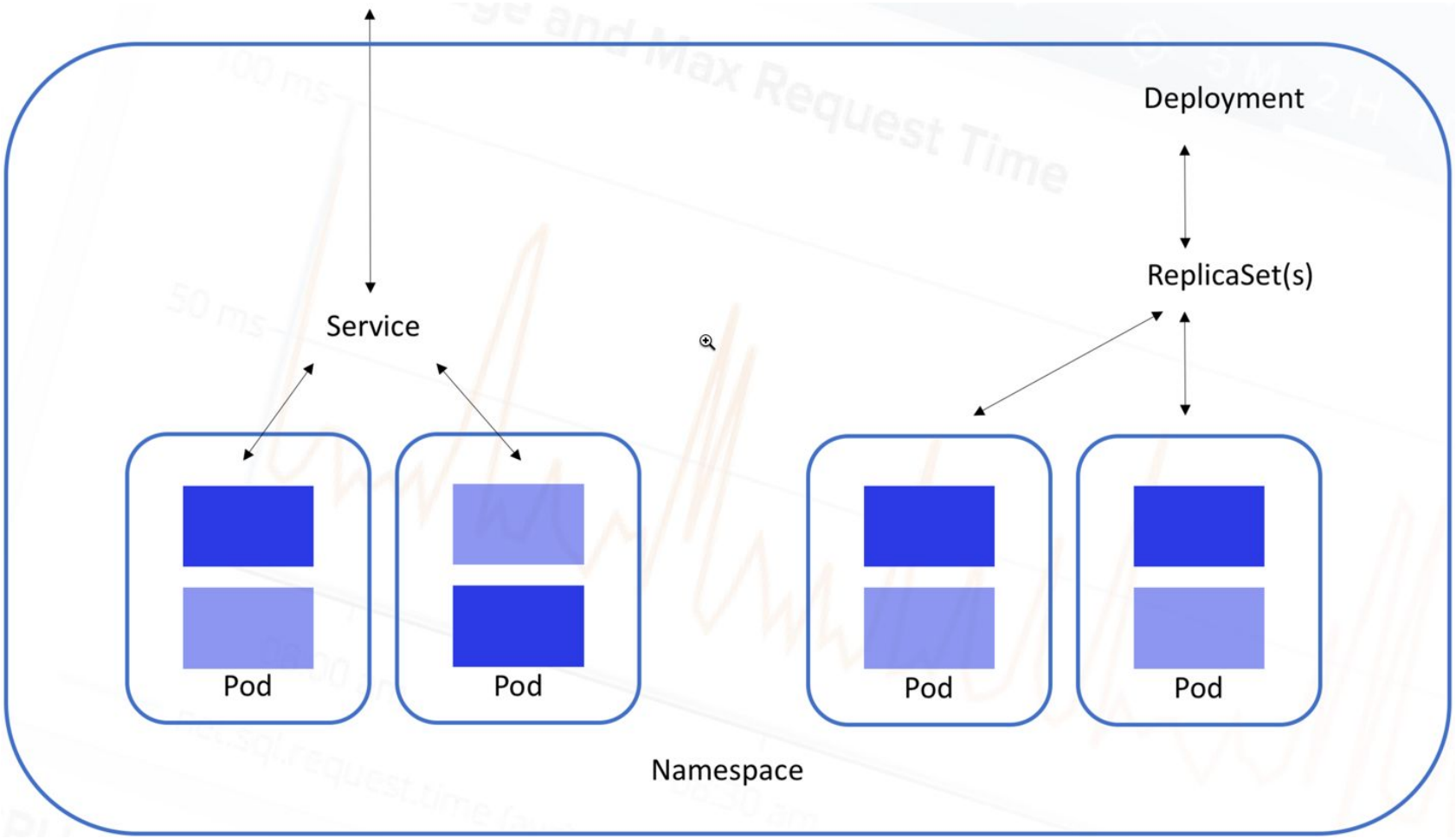
- Pods Priority and Preemption
  - Kubernetes will use value on object “PriorityClass” for consideration. Pods will reference this for define

```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

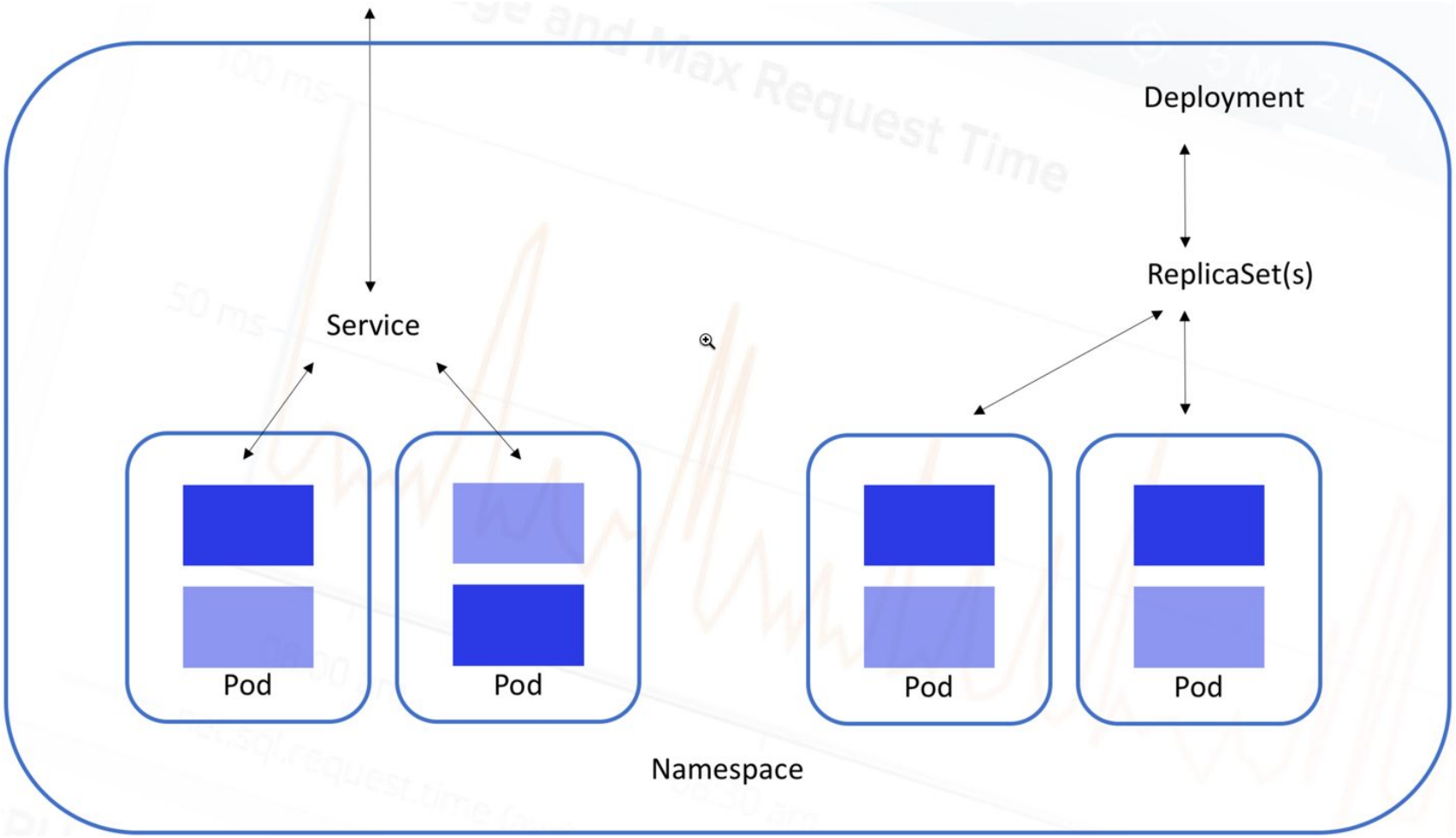
Apps | NMac Ked - Mac O... | Medium | jerkins | vagrant | Mesos | Vue | doc

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)

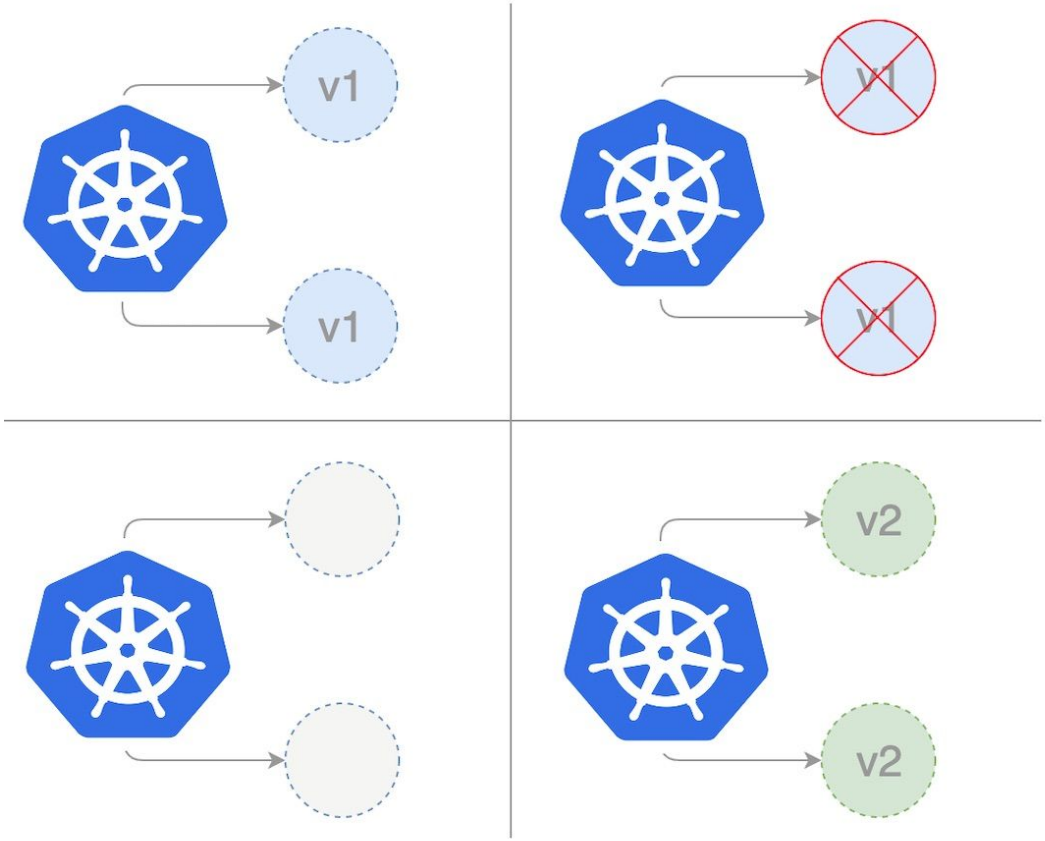
Deployment/RS and Update

- 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

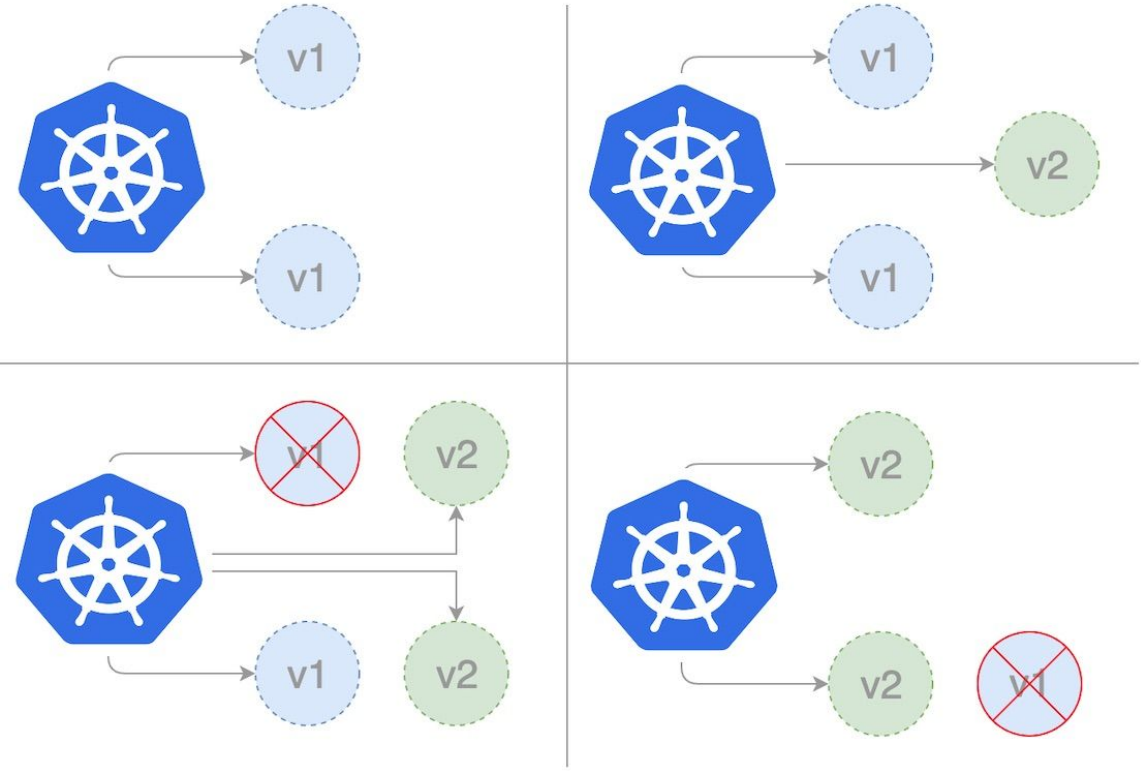


webtest_deploy.yml

```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)



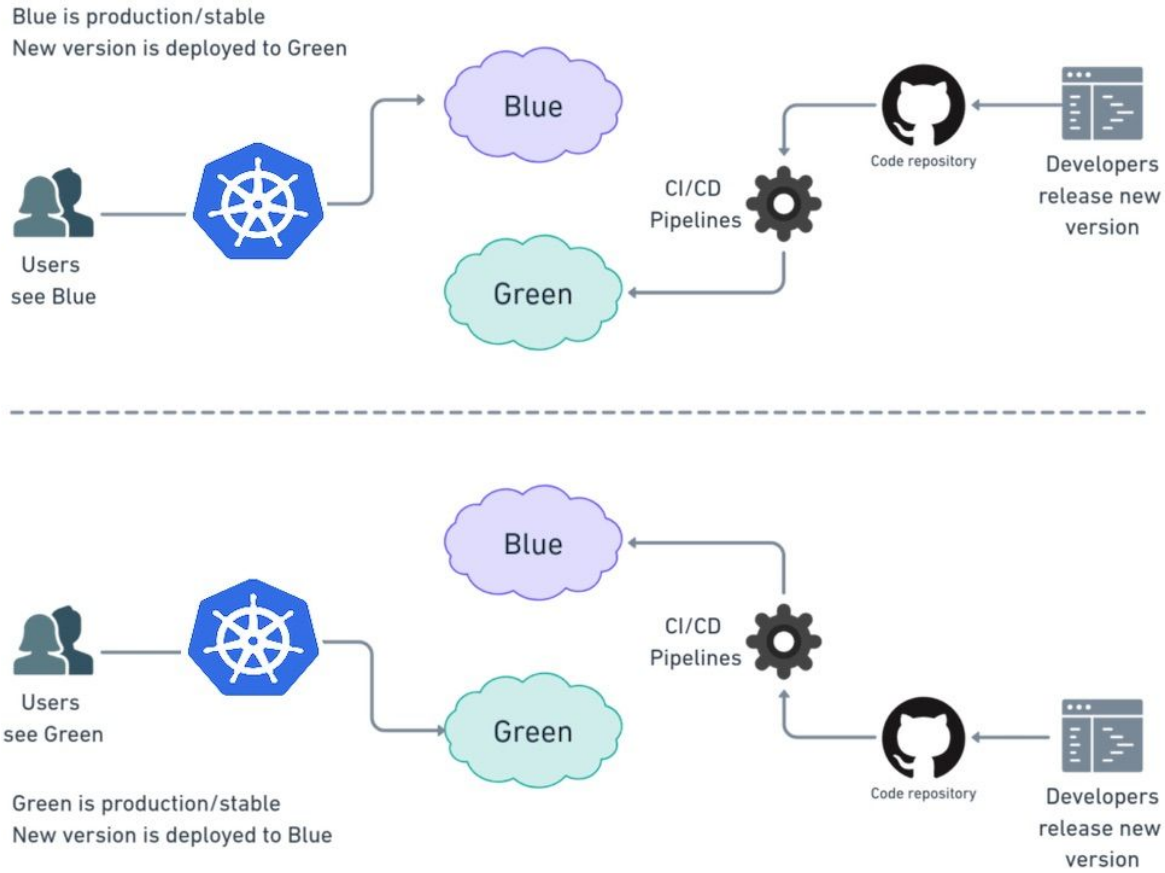
webtest_deploy.yml

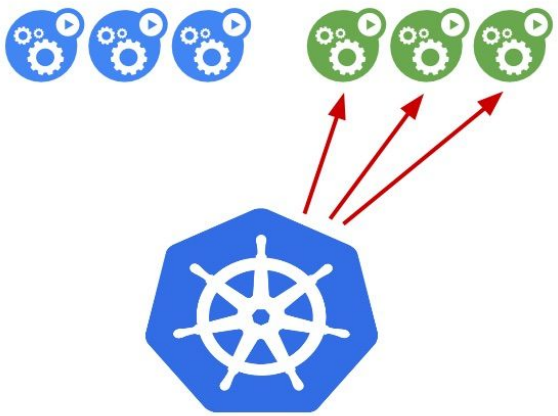
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 4
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavaible: 1
      maxSurge: 1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

kubernetes by Google

Deployment/RS and Update

- Blue/Green



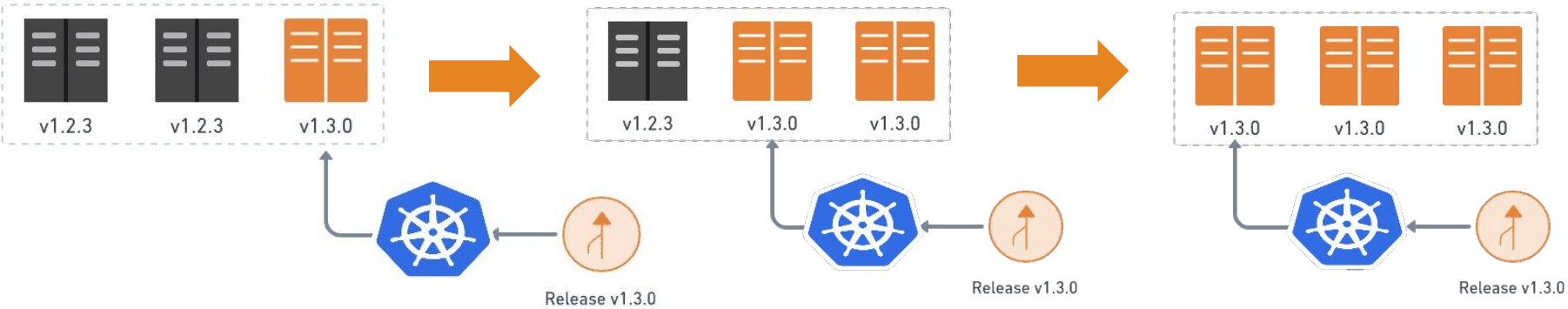


webtest_deploy.yml webtest_svc.yml

```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
```

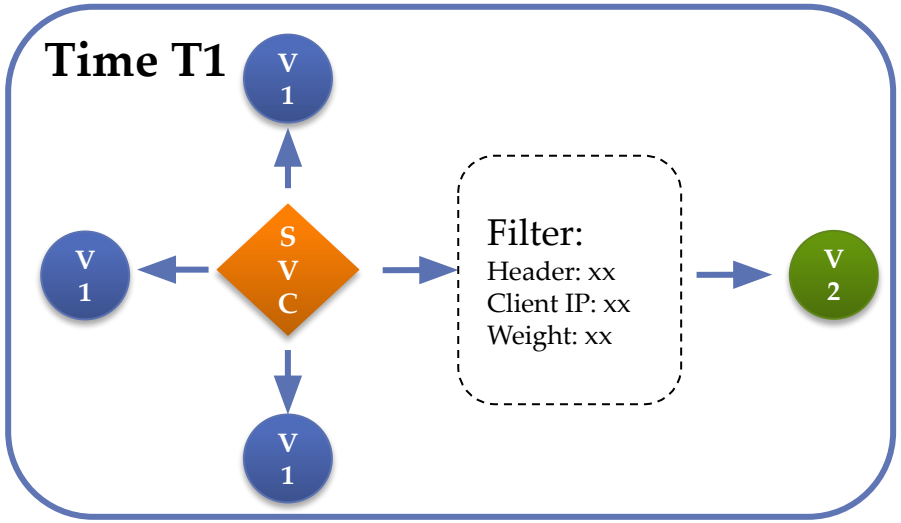
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest_V2
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
```

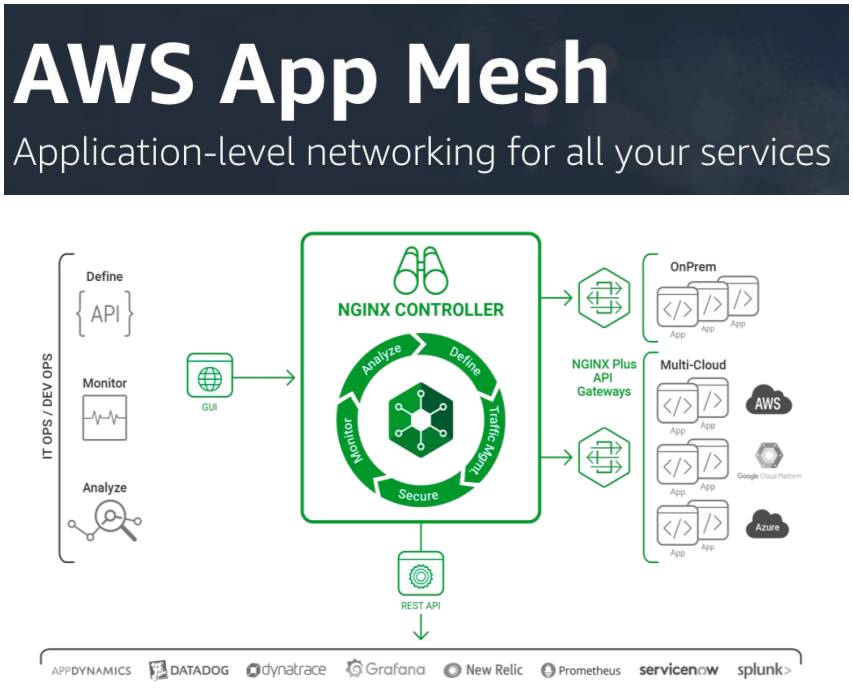
webtest_deploy.yml ● webtest_svc.yml ×

```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.

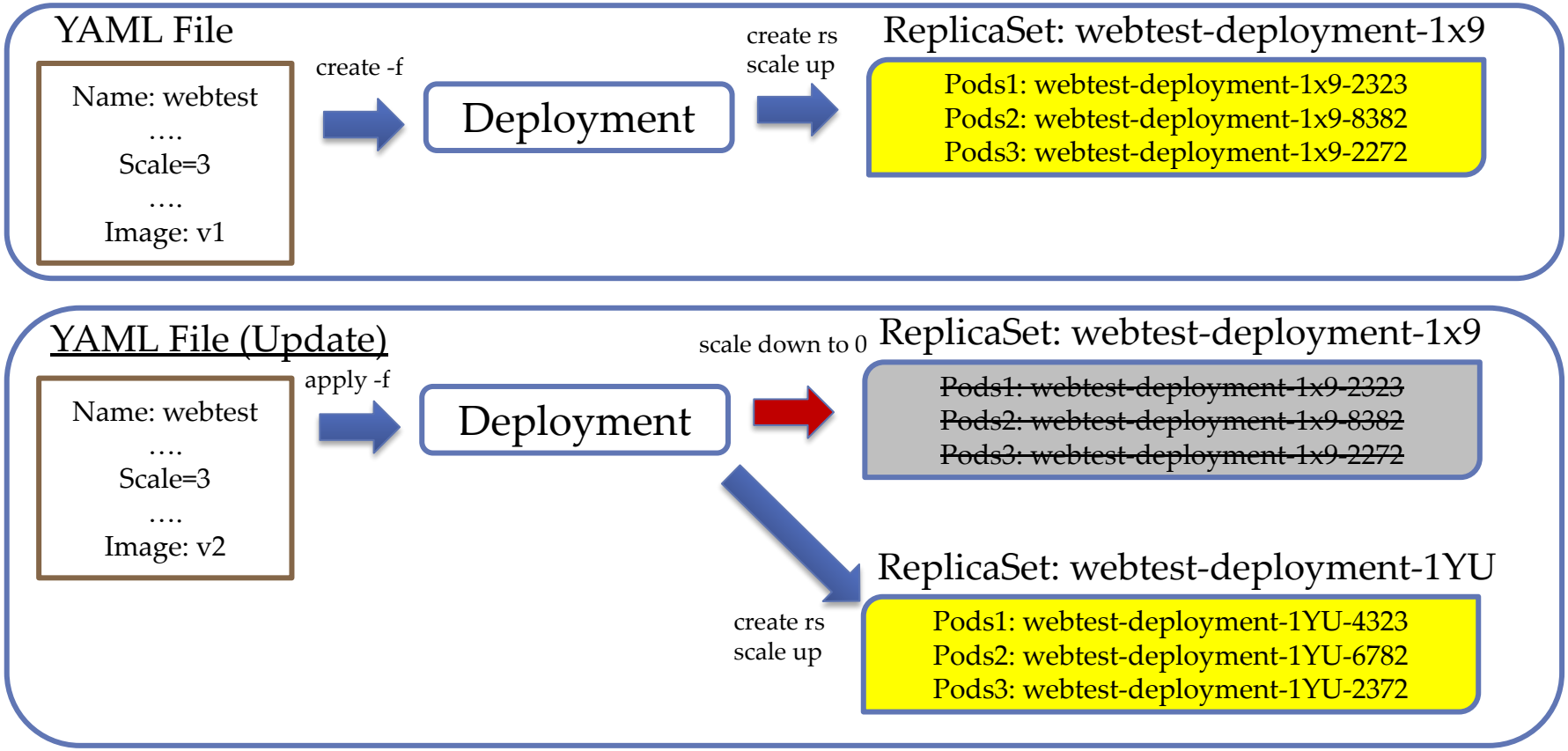
vamp

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

Deployment/RS and Update

- Deployment update strategy
  - Rollout History

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

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

Deployment/RS and Update

- Deployment update strategy
  - Rollback process

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

  - Rollout History

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

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

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**
  - gcePersistent
  - awsElasticBlockStore
  - Nfs
  - Iscsi
  - fc(fibrechannel)
  - Flocker
  - Glusterfs
  - Rbd
  - Cephfs
  - gitRepo (deprecated)
  - Secret
  - ***persistentVolumeClaim (PVC) (Talk Day2)***
  - downwardAPI
  - Projected
  - azureDisk
  - azureFile
  - vsphereVolume
  - Quobyte
  - PortworxVolume
  - ScaleIO
  - StorageOS
  - Local
  - ConfigMap
  - Secret
  - csi

Volume

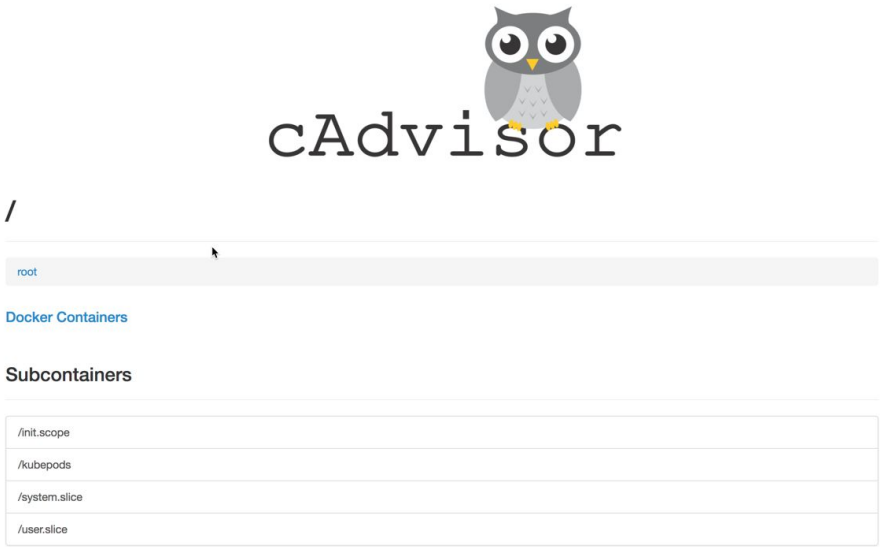
- hostPath:
  - Mount file/directory on host to Pods
  - Recommend for Pods that need read some data on host path (Such as cAdvisor etc)
  - Consideration
    - hostPath may effect to some of path/file not equal on different host
    - Kubernetes resource scheduling cannot check readiness of hostPath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

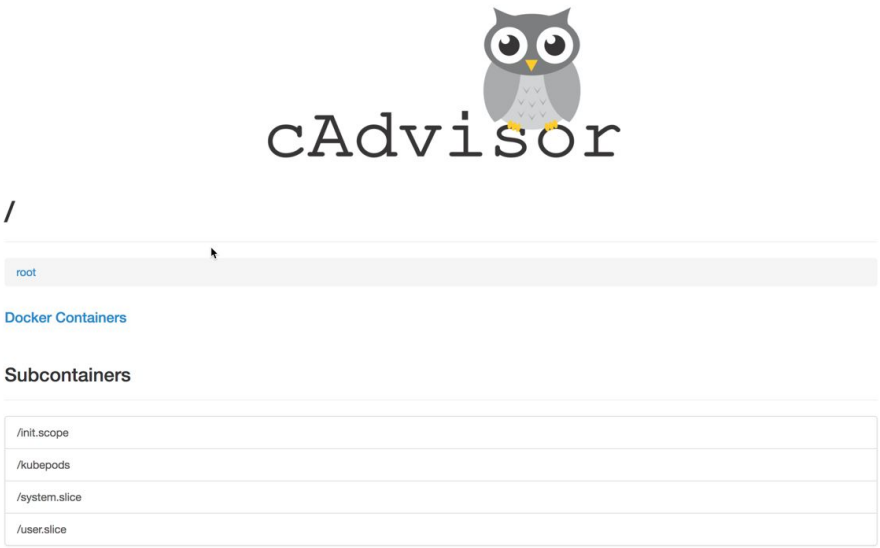
```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      ports:
        - containerPort: 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 #
```

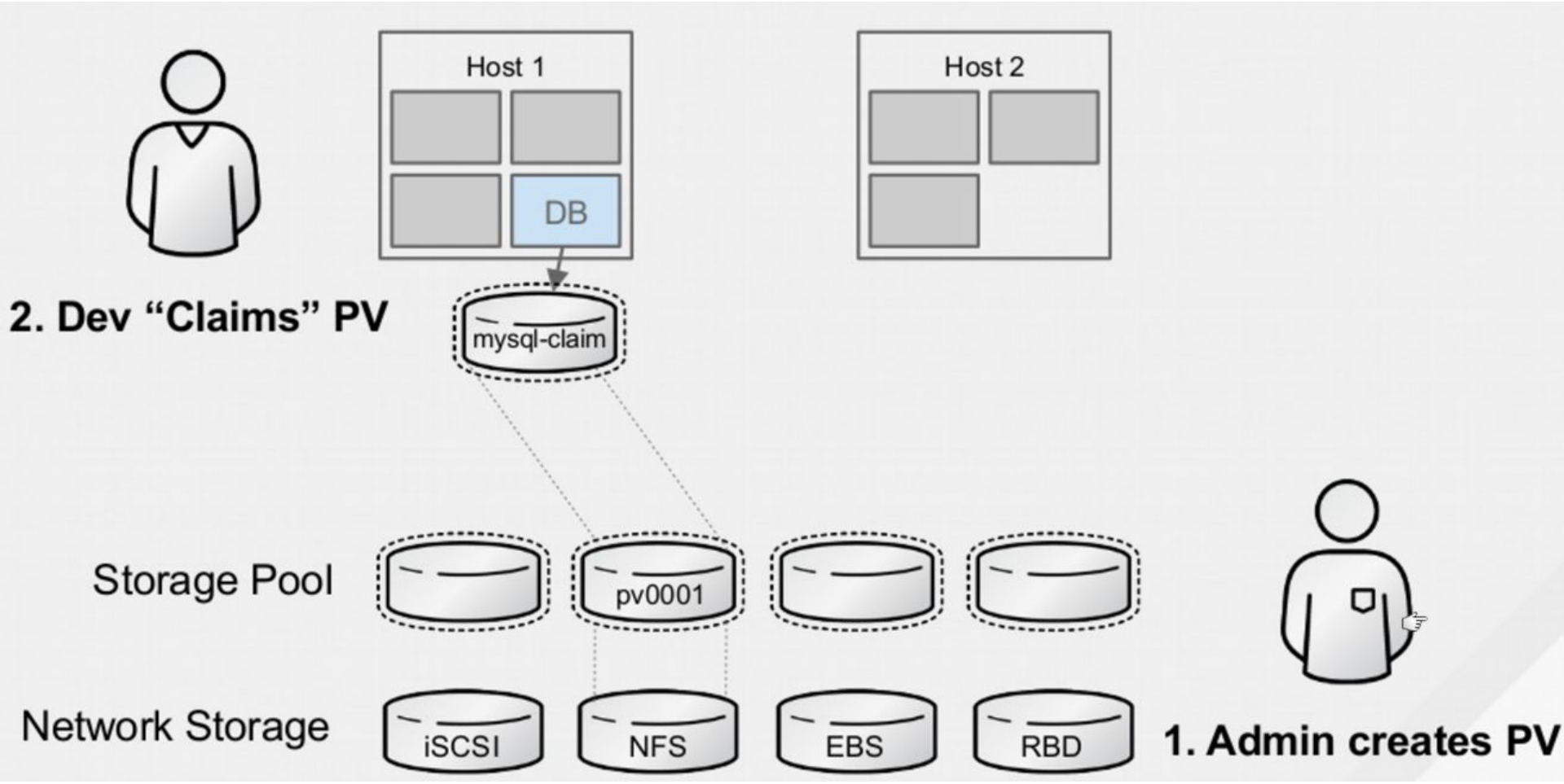
Workshop: Volume

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 and Readiness Probe

Liveness and Readiness Probe

- Pods are you still alright ?
  - Normally kubernetes will check Pods status and process criteria for orchestrator following “restartPolicy”
    - Always (Default)
    - On-Failure
    - None
  - Pods status is depend on “ContainerState” in that Pods (1-M)
    - Waiting (ContainerStateWaiting)
    - Running (ContainerStateRunning)
    - Terminated (ContainerStateTerminated)

Liveness and Readiness 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 and Readiness 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)
    - TCPSocketAction: Start TCP communication on specific port of container (expect result: port open)
    - HTTPGetAction: Send http/https request to specific port and path (expect result: return 200 for OK)
  - Probe's result
    - Success
    - Failed
    - Unknown
  - livenessProbe and readinessProbe Difference
    - livenessProbe: check status of container is running properly or not ?
    - readinessProbe: check readiness for process service or not ?

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) 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
```

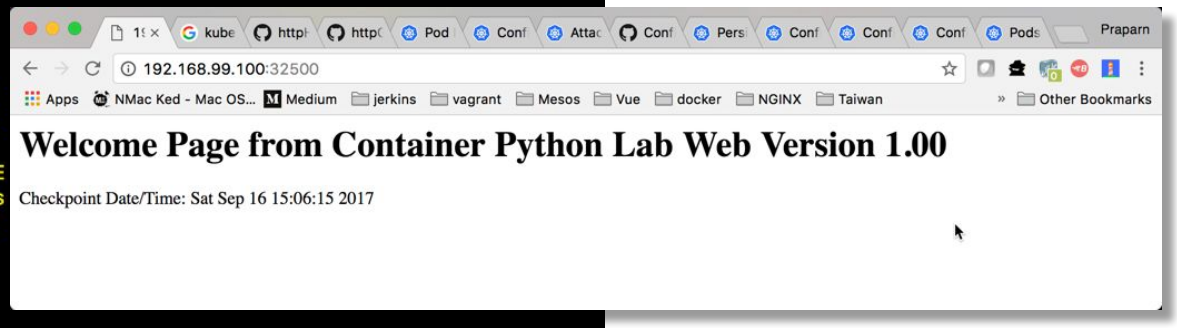
```yaml
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - containerPort: 5000
            protocol: TCP
          readinessProbe:
            exec:
              command:
                - cat
                - /usr/src/app/main.py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

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

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:

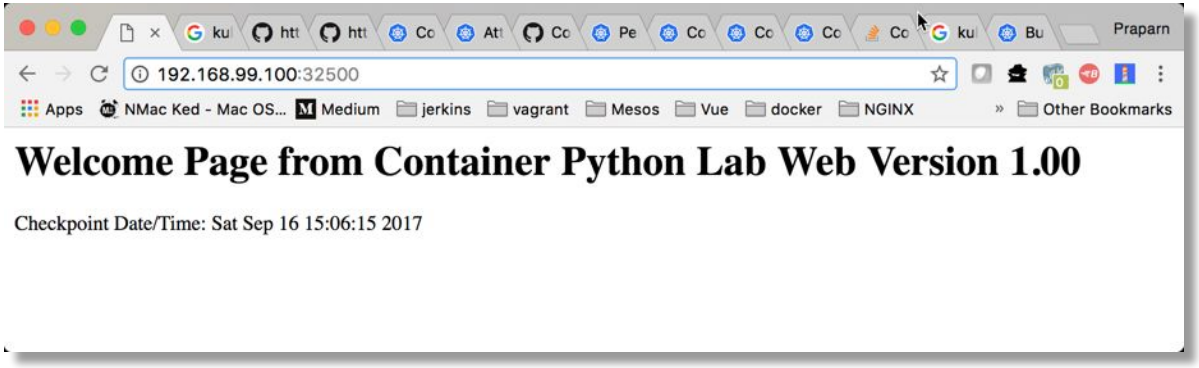
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
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

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

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"...
```

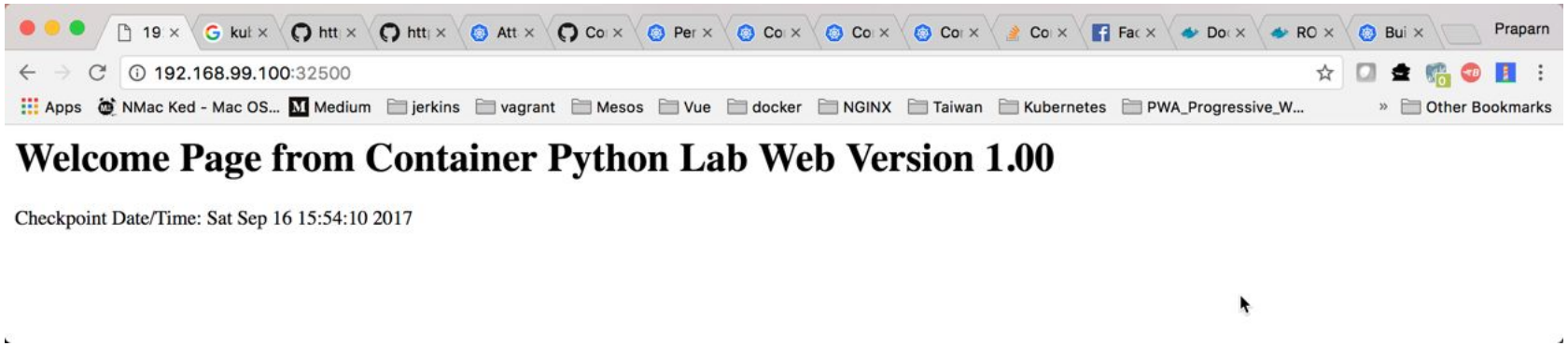
Liveness and Readiness Probe

- 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
        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
        initialDelaySeconds: 15
        periodSeconds: 10
        timeoutSeconds: 15
        successThreshold: 1
        failureThreshold: 3
```

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
...
```

Workshop: Liveness and Readiness

192.168.99.100:32500

Welcome Page from Container Python Lab Web Version 1.00

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

Runner | Import | Builder | Team Library | IN SYNC | praparn

Filter

History | Collections

Clear all

Today

GET http://192.168.99.100:32500

September 8

GET http://192.168.99.100/users/removeuser/99

GET http://192.168.99.100/users/1

POST http://192.168.99.100/users/insertuser

GET http://192.168.99.100/init

GET http://192.168.99.100/

August 27

http://192.168.99.100

No Environment

GET http://192.168.99.100:32500 Params Send Save

Authorization | Headers | Body | Pre-request Script | Tests | Code

Type: No Auth

Body | Cookies | Headers (4) | Tests

Status: 200 OK Time: 46 ms

content-length → 113

content-type → text/html; charset=utf-8

date → Sat, 16 Sep 2017 15:56:21 GMT

server → Werkzeug/0.12.2 Python/2.7.13

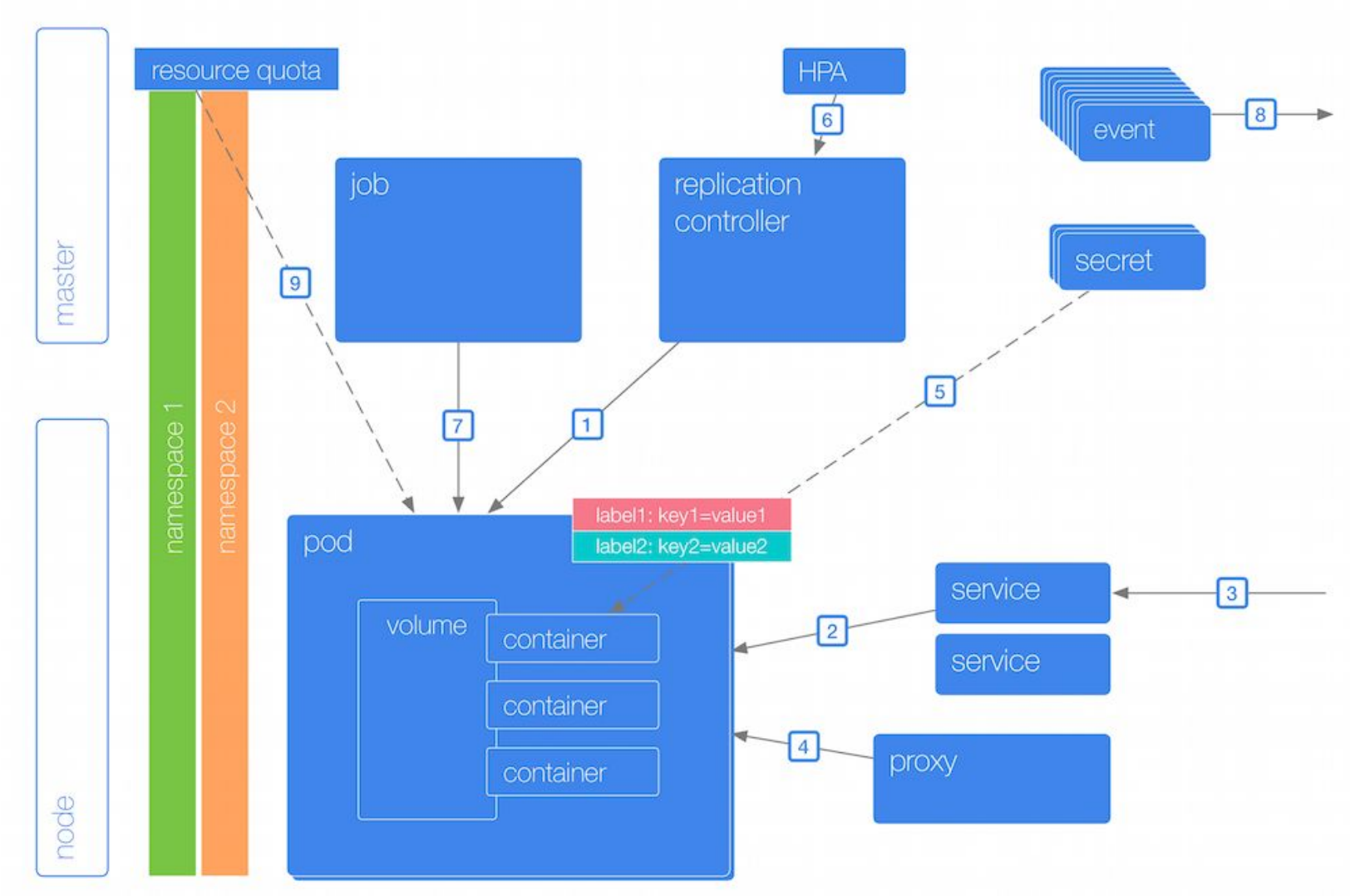
kubernetes by Google

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

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"
        limits:
          memory: "32Mi"
          cpu: "1"
      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
  - Disk I/O
    - PersistenctVolumeClaims
    - Request.Storage
  - Counting Object
    - Pods
    - Service
    - Replication Controller
    - 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
```

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

Resource Management and HPA

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

```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
```

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_v1
      resources:
        requests:
          memory: "16Mi"
          cpu: "150m"
        limits:
          memory: "16Mi"
          cpu: "150m"
      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) and automatic trigger deployment to scale-up application

kubectl autoscale <type/name> <option:>

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

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
```

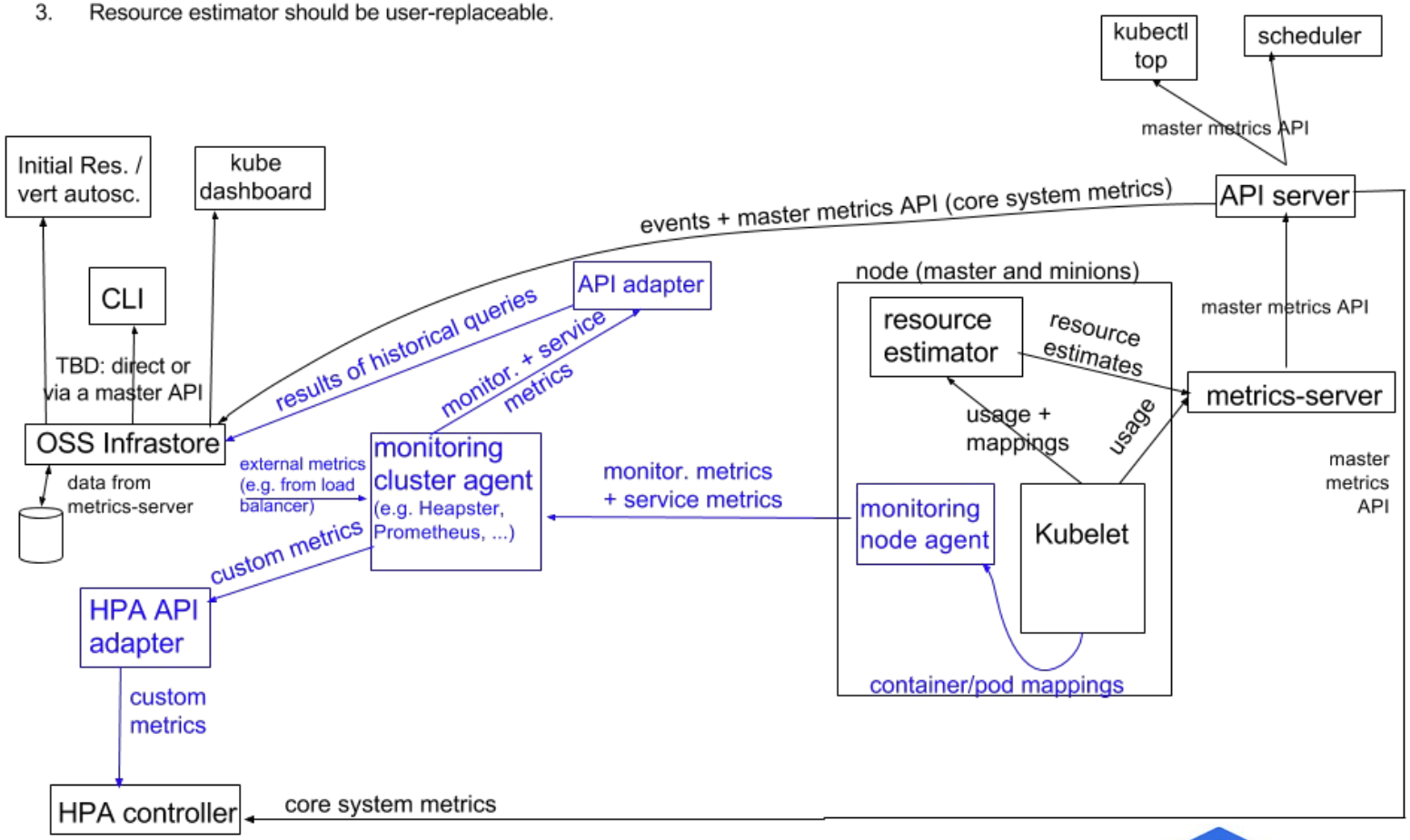
```yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  annotations:
  name: webtest
  namespace: default
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: webtest
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: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

```yaml
apiVersion: 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
```

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

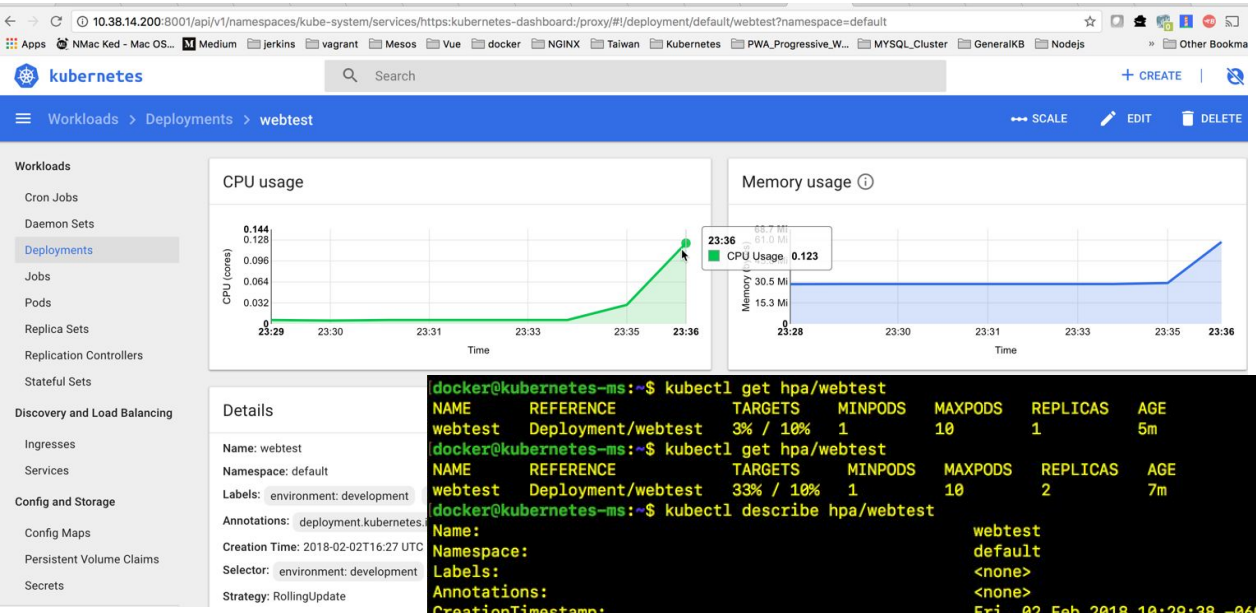
- Generate load by busybox (wget every 10 ms)

```
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:~$
```

Recap Day 3

- 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)

Q&A

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

DAY 4

Outline Day 4

- Fundamental of Kubernetes
  - Configmap and Secret
  - 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

ConfigMap and Secret

ConfigMap and Secret

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

ConfigMap and Secret

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

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
```

ConfigMap and Secret

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

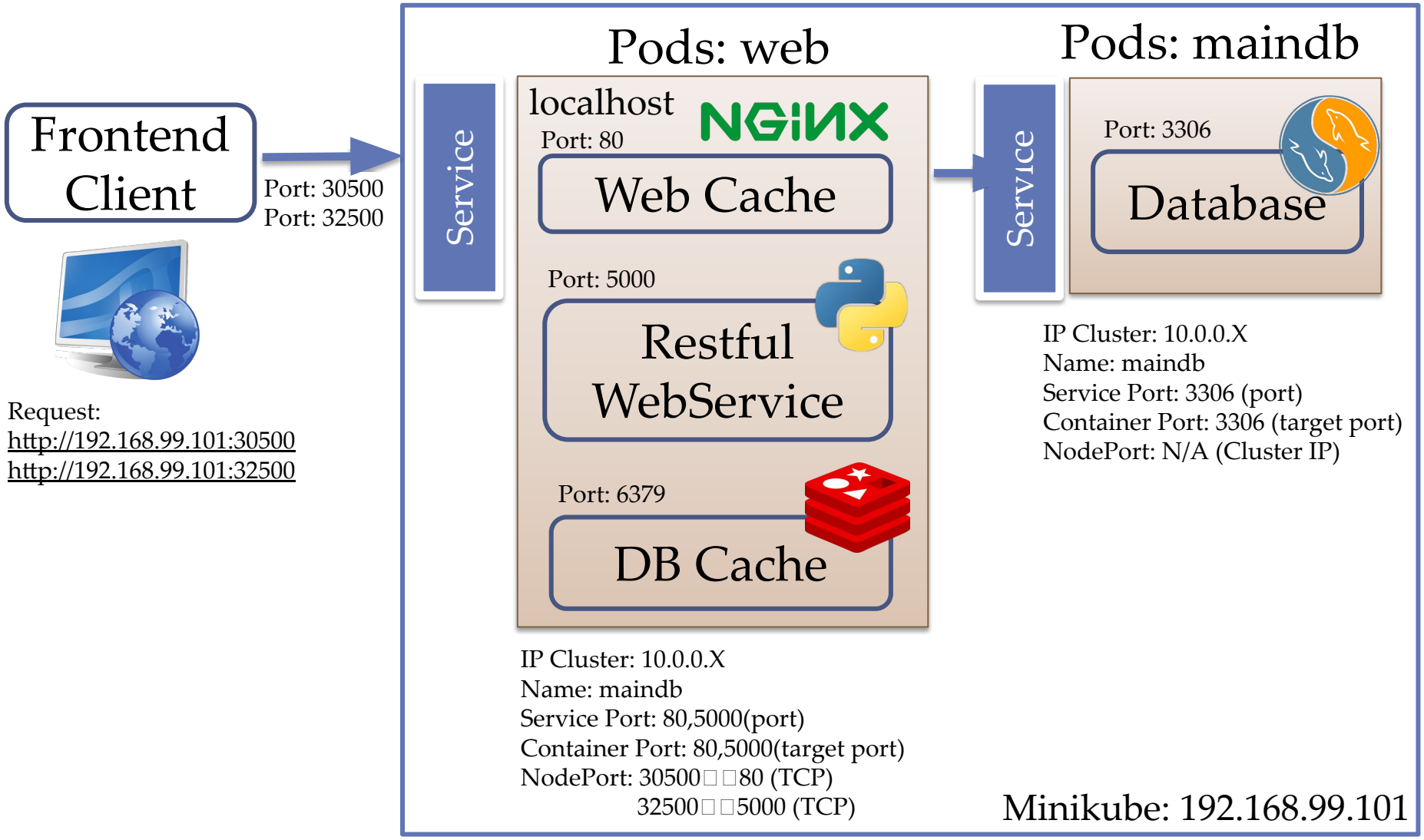
kubectl create secret generic <name> <option>

- 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=
```

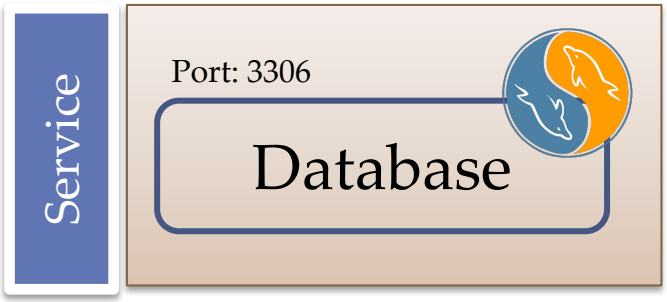
ConfigMap and Secret



Kubernetes: Production Workload Orchestration

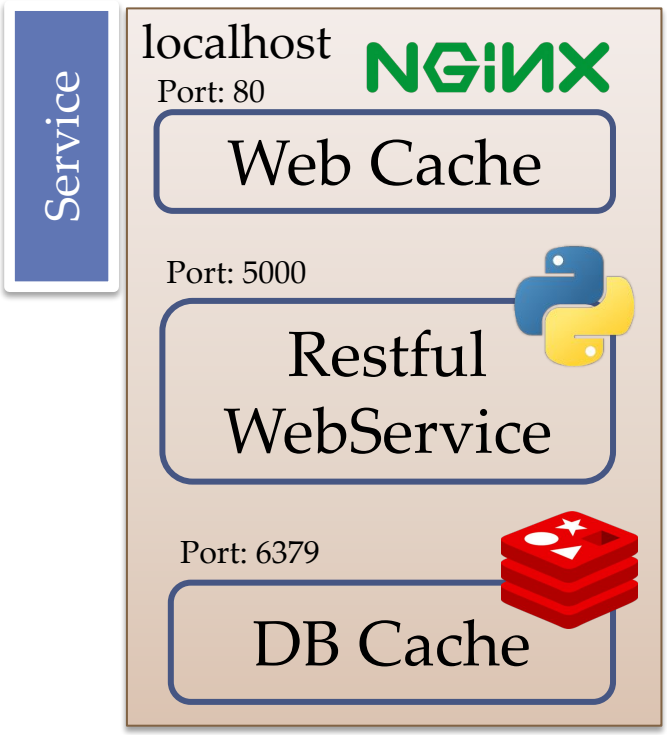
kubernetes by Google

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

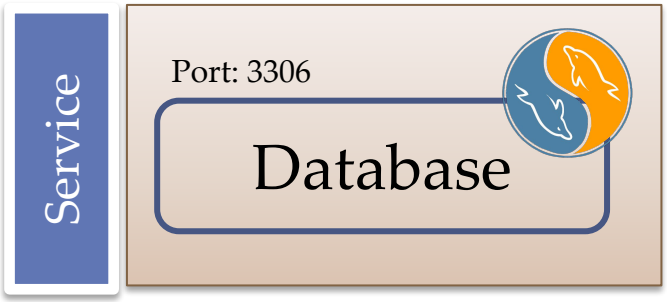
```python
CACHE_DB = redis.Redis(host=os.environ.get('REDIS_HOST', 'cachedb'), port=6379)
db = MySQLdb.connect("maindb","root","password")
MAIN_DB = db.cursor()
```

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 Day 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 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

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

Day 2

Outline 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

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>

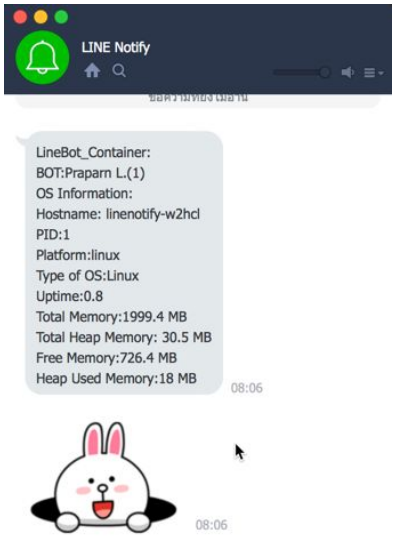
Job and Cron Job

Job.yml:

WorkShop_2.2_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
        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: "EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX"
      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>
```



Job and Cron Job

kubernetes

Reference

HOME GETTING STARTED CONCEPTS TASKS TUTORIALS REFERENCE CONTRIBUTE

Reference

- Standardized Glossary
- Kubernetes Issues and Security
- Using the Kubernetes API
- Accessing the API
- API Reference
- Setup tools reference
  - Kubeadm
    - Overview of kubeadm
    - kubeadm init
    - kubeadm join

kubeadm init

This command initializes a Kubernetes control-plane node.

- **What's next**

Run this command in order to set up the Kubernetes control plane

Synopsis

Run this command in order to set up the Kubernetes control plane

The "init" command executes the following phases:

Options

```
--apiserver-advertise-address string   The IP address the API Server will advertise it's listening on. If not set the default netw
--apiserver-bind-port int32            Port for the API Server to bind to. (default 6443)
--apiserver-cert-extra-sans strings    Optional extra Subject Alternative Names (SANs) to use for the API Server serving certifica
--cert-dir string                      The path where to save and store the certificates. (default "/etc/kubernetes/pki")
--certificate-key string               Key used to encrypt the control-plane certificates in the kubeadm-certs Secret.
--config string                        Path to a kubeadm configuration file.
--cri-socket string                    Path to the CRI socket to connect. If empty kubeadm will try to auto-detect this value; use
--dry-run                              Don't apply any changes; just output what would be done.
--feature-gates string                 A set of key=value pairs that describe feature gates for various features. No feature gates
--help                                 help for init
--ignore-preflight-errors strings      A list of checks whose errors will be shown as warnings. Example: 'IsPrivilegedUser,Swap'.
--image-repository string              Choose a container registry to pull control plane images from (default "k8s.gcr.io")
--kubernetes-version string            Choose a specific Kubernetes version for the control plane. (default "stable-1")
--node-name string                     Specify the node name.
--pod-network-cidr string              Specify range of IP addresses for the pod network. If set, the control plane will automatic
--service-cidr string                  Use alternative range of IP address for service VIPs. (default "10.96.0.0/12")
--service-dns-domain string            Use alternative domain for services, e.g. "myorg.internal". (default "cluster.local")
--skip-certificate-key-print           Don't print the key used to encrypt the control-plane certificates.
--skip-phases strings                  List of phases to be skipped
--skip-token-print                     Skip printing of the default bootstrap token generated by 'kubeadm init'.
--token string                         The token to use for establishing bidirectional trust between nodes and control-plane nodes
--token-ttl duration                   The duration before the token is automatically deleted (e.g. 1s, 2m, 3h). If set to '0', th
--upload-certs                         Upload control-plane certificates to the kubeadm-certs Secret.
```

savitaashture commented on Dec 13, 2018 — Author

Thanks a lot.

Its working now.
Earlier i haven't set --feature-gates=TTLAfterFinished=true in controller yaml thats the reason it was not working.

3

TTL Mechanism for Finished Jobs

FEATURE STATE: `Kubernetes v1.12` alpha

Another way to clean up finished Jobs (either `Complete` or `Failed`) automatically is to use a TTL mechanism provided by a TTL controller for finished resources, by specifying the `.spec.ttlSecondsAfterFinished` field of the Job.

When the TTL controller cleans up the Job, it will delete the Job cascadingly, i.e. delete its dependent objects, such as Pods, together with the Job. Note that when the Job is deleted, its lifecycle guarantees, such as finalizers, will be honored.

For example:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pi-with-ttl
spec:
  ttlSecondsAfterFinished: 100
  template:
    spec:
      containers:
      - name: pi
        image: perl
        command: ["perl",  "-Mbignum=bpi", "-wle", "print bpi(2000)"]
      restartPolicy: Never
```

The Job `pi-with-ttl` will be eligible to be automatically deleted, `100` seconds after it finishes.

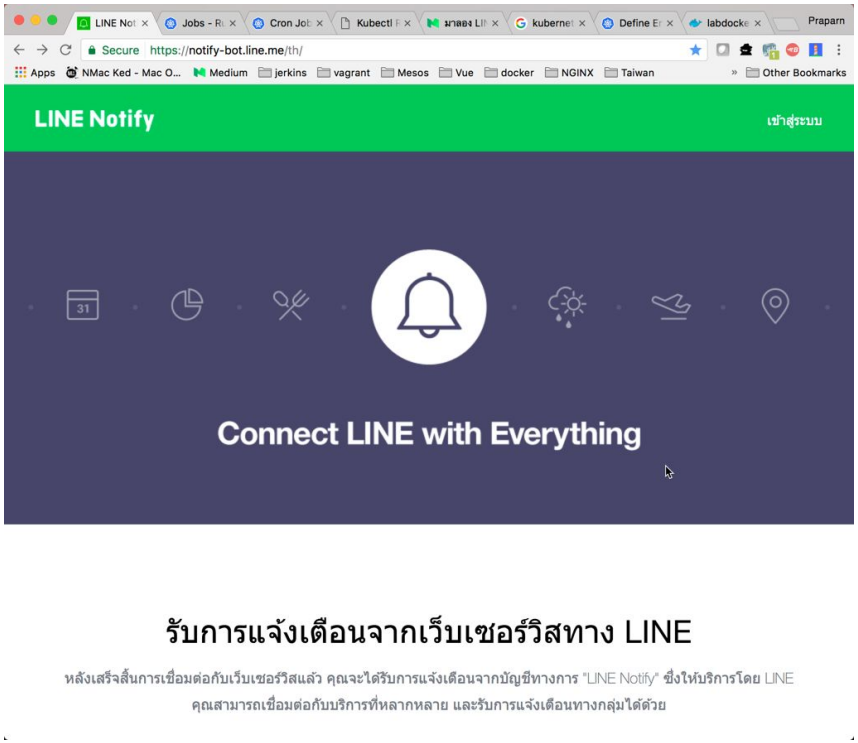
https://kubernetes.io/docs/reference/setup-tools/kubeadm/kubeadm-init/

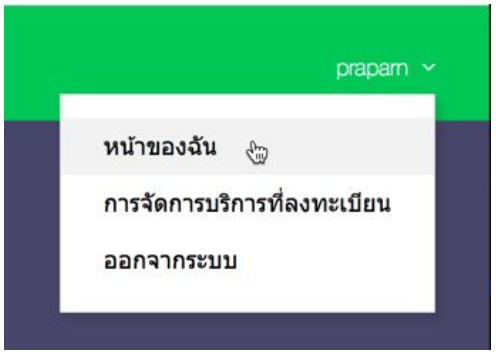
https://github.com/kubernetes/kubernetes/issues/71993

kubernetes by Google

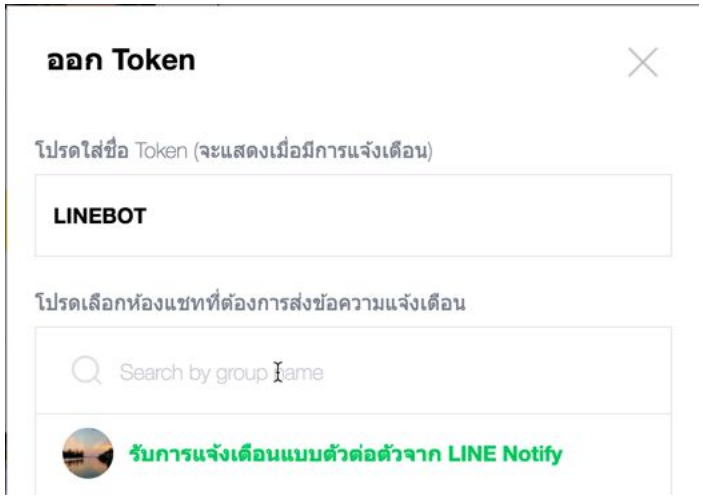
Workshop: Job and CronJob

- 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.

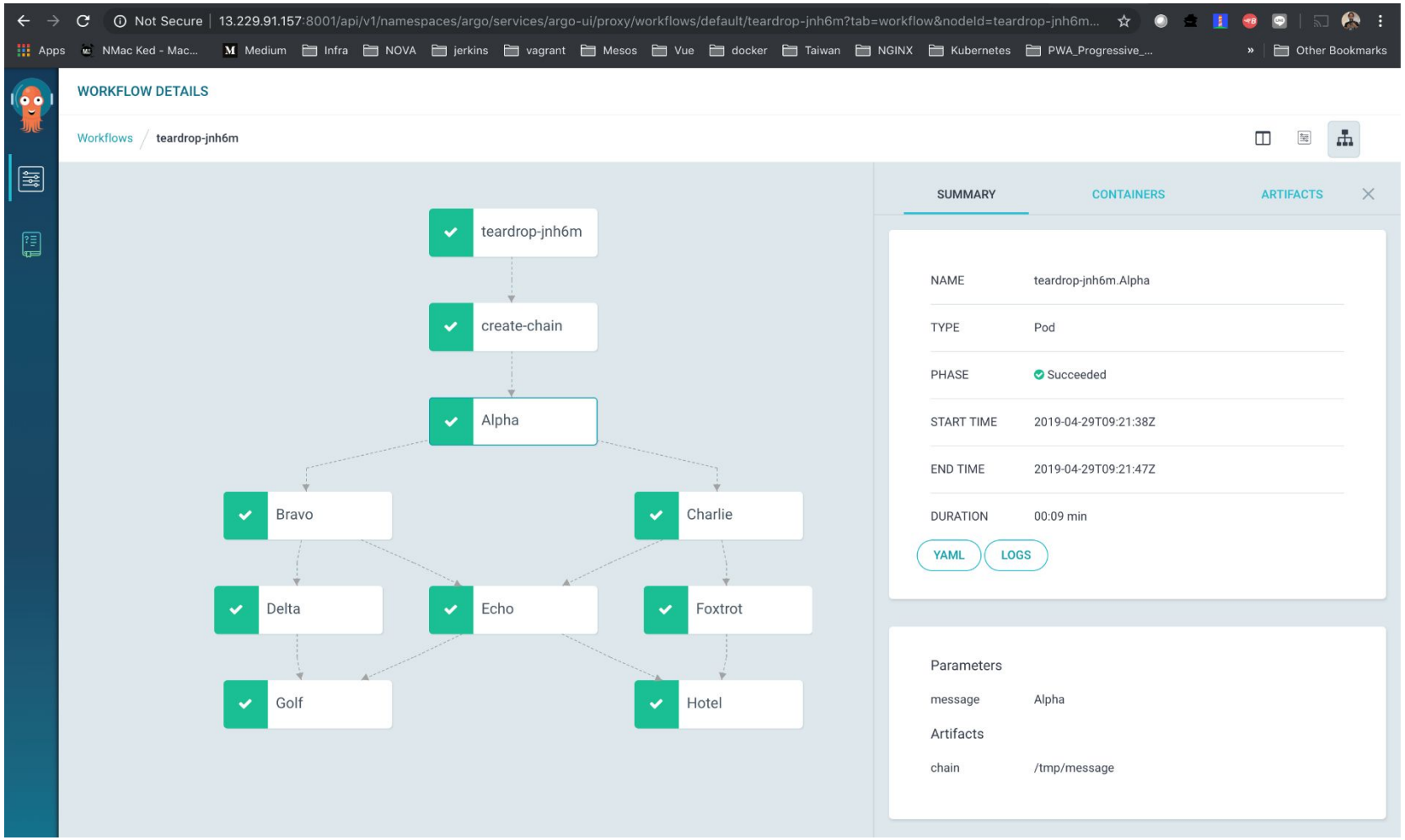
- Argo Workflows - Container-native Workflow Engine
- Argo CD - Declarative GitOps Continuous Delivery
- Argo Events - Event-based Dependency Manager

https://github.com/argoproj/argo

kubernetes by Google

Job and Cron Job

- WorkFlow Job



https://github.com/argoproj/argo

Debug Log and Monitoring

Debug Log and Monitoring

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

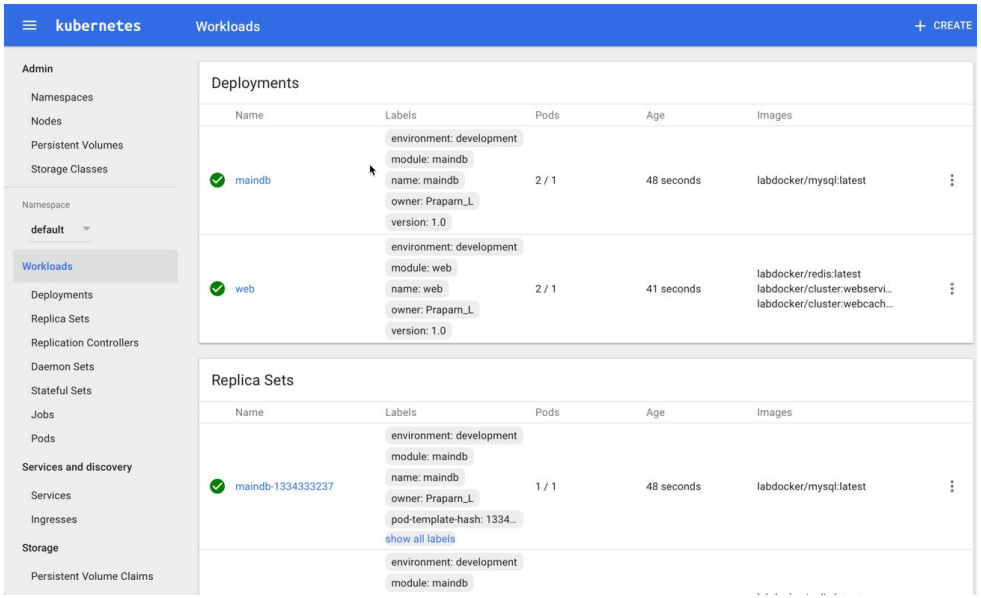
```
kubectl logs pods/<pods name> -c <container name> <option>
```

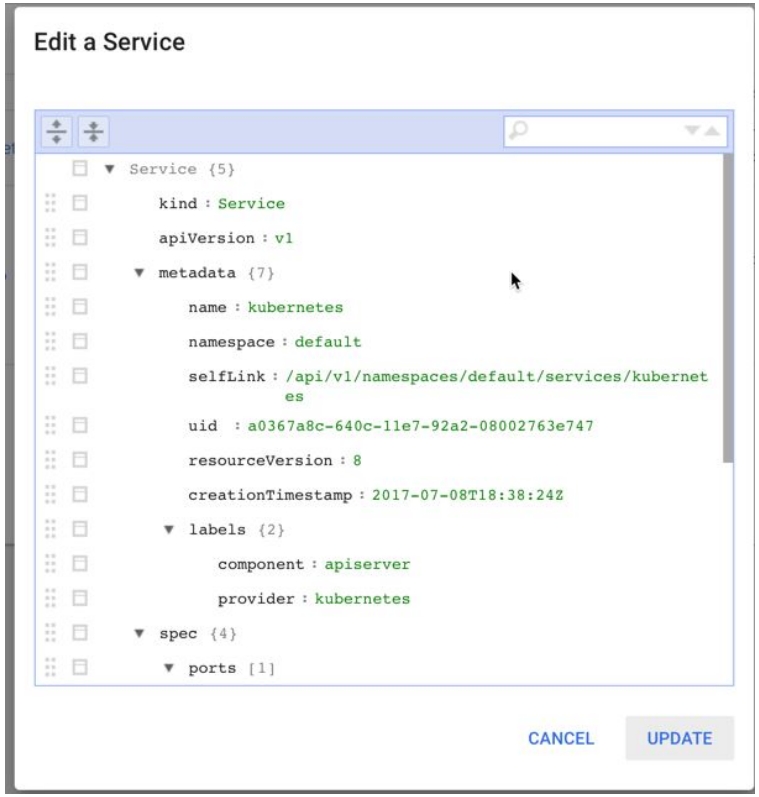
- Option:
  - -p, --previous=false ==> Printout last container fail in Pods
  - -f, --follow=false ==> Follow stream log
  - -c, --container ==> Specific container for check log
  - -l, --selector ==> Select filter some label for check log
- Ex:
  - kubectl logs -f pods/maindb –c maindb

Debug Log and Monitoring

- Monitoring kubernetes system via dashboard

kubernetes dashboard





Debug Log and Monitoring

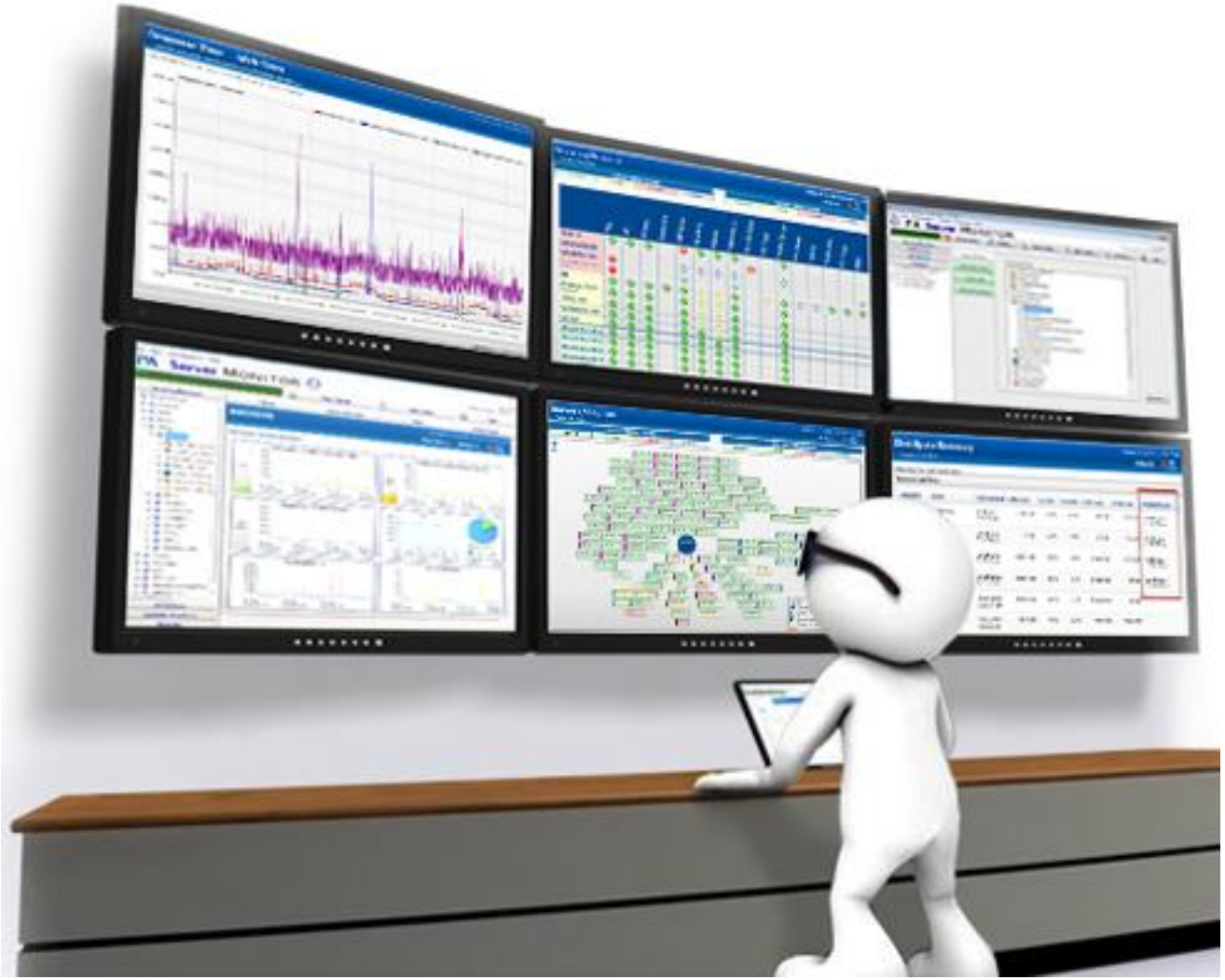
- Ephemeral Container (Obsolete):
  - Getting debug application inside "Pods"
  - Ideally is run container inside pods for operate as "debugger"
  - Not necessary for external component

API changes and improvements for ephemeral containers

The API used to create Ephemeral Containers changes in 1.22. The Ephemeral Containers feature is alpha and disabled by default, and the new API does not work with clients that attempt to use the old API.

For stable features, the kubectl tool follows the Kubernetes version skew policy; however, kubectl v1.21 and older do not support the new API for ephemeral containers. If you plan to use `kubectl debug` to create ephemeral containers, and your cluster is running Kubernetes v1.22, you cannot do so with kubectl v1.21 or earlier. Please update kubectl to 1.22 if you wish to use `kubectl debug` with a mix of cluster versions.

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.

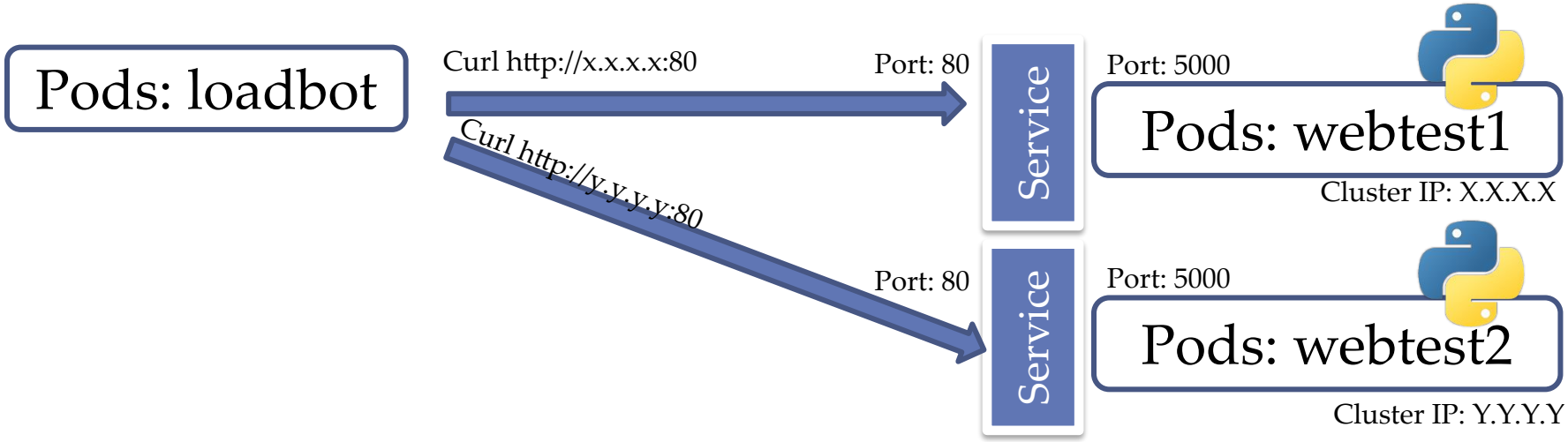
Additional controllers

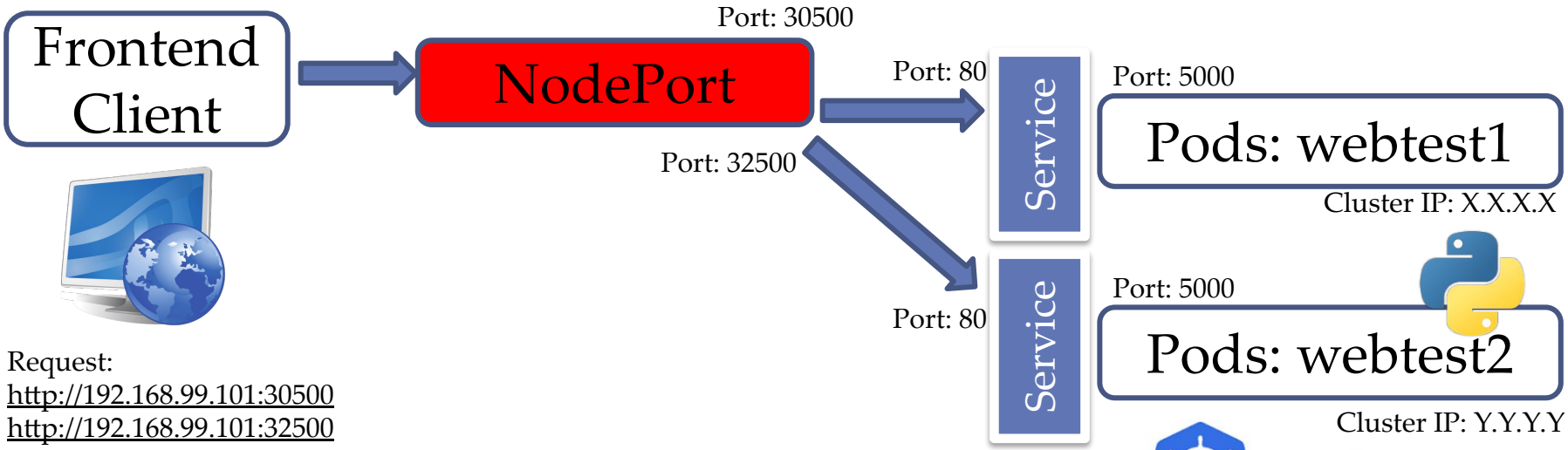
> **Caution:** This section links to third party projects that provide functionality required by Kubernetes. The Kubernetes project authors aren't responsible for these projects. This page follows CNCF website guidelines by listing projects alphabetically. To add a project to this list, read the content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

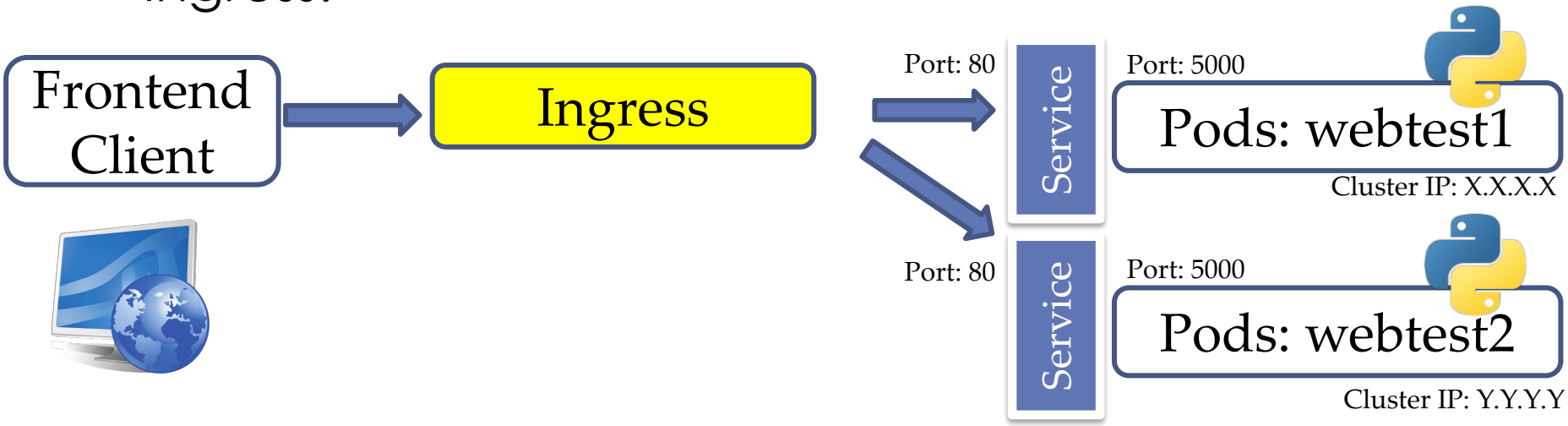
- Service:





Ingress Network

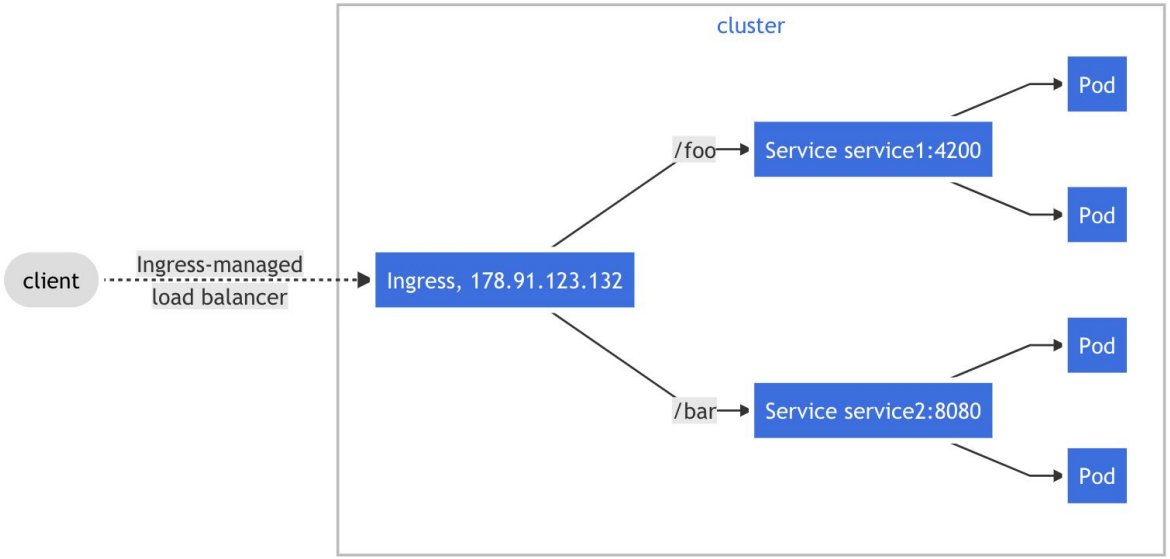
- Ingress:



Ingress Network

Users > praparnlueangphoonlap > Desktop >

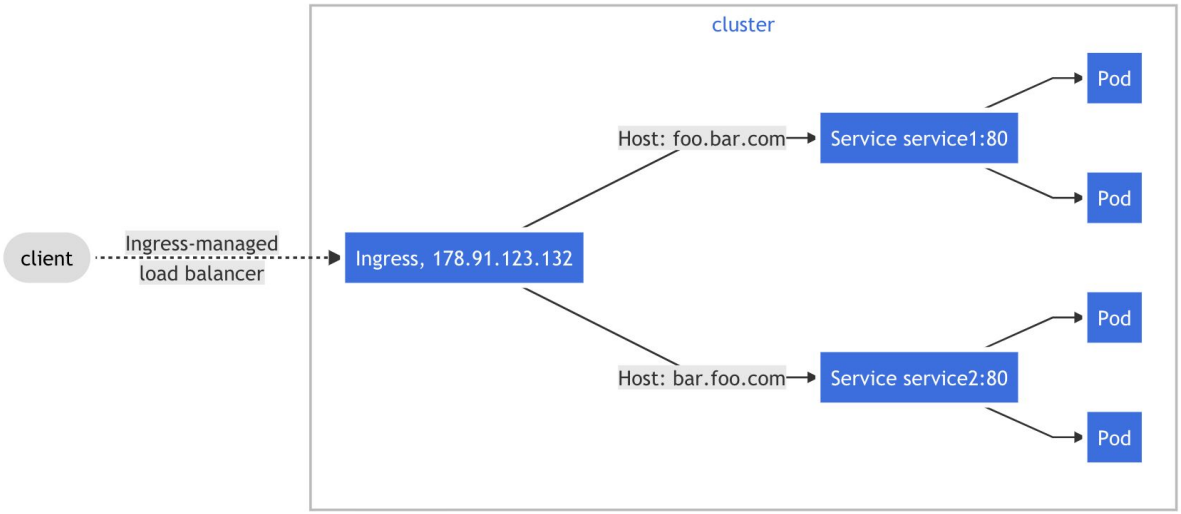
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: simple-fanout-example
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - path: /foo
        pathType: Prefix
        backend:
          service:
            name: service1
            port:
              number: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



Ingress Network

Users > praparnlueangphoonlap > Desktop > !

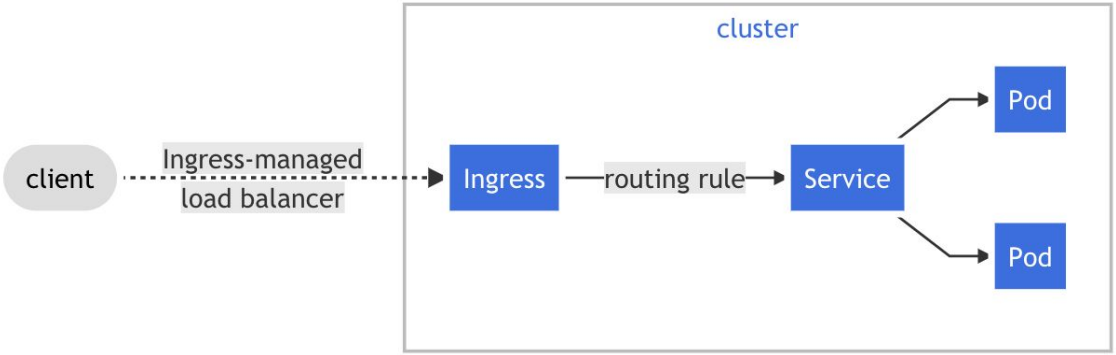
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: name-virtual-host-ingress
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: service1
            port:
              number: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

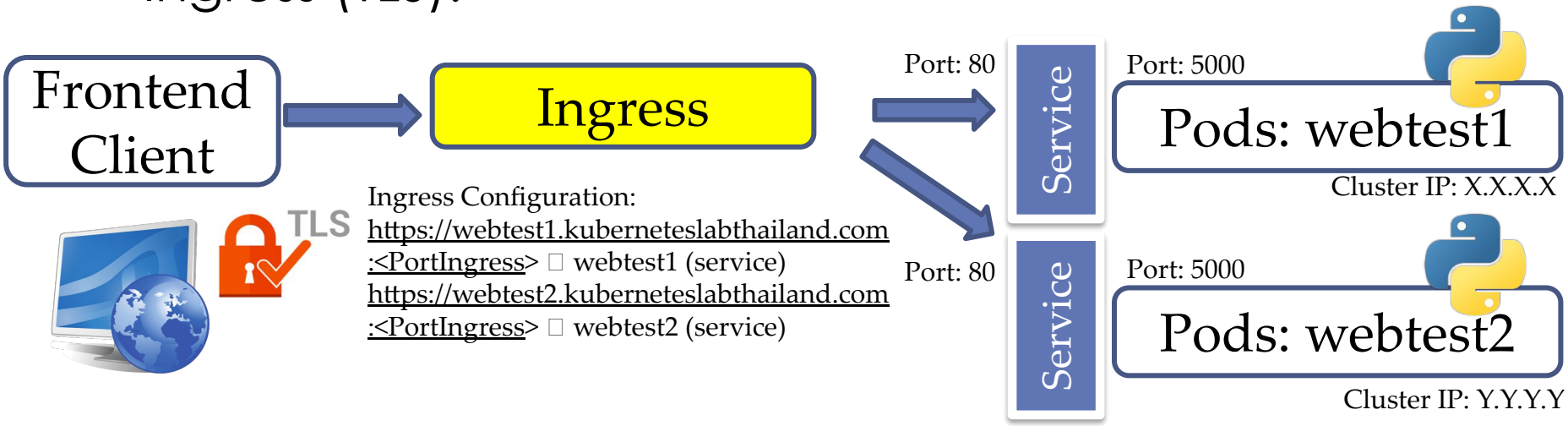
- Multiple-controller of ingress
  - Kubernetes have capability for housing multiple ingress controller on same cluster
  - Each ingress controller

- 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

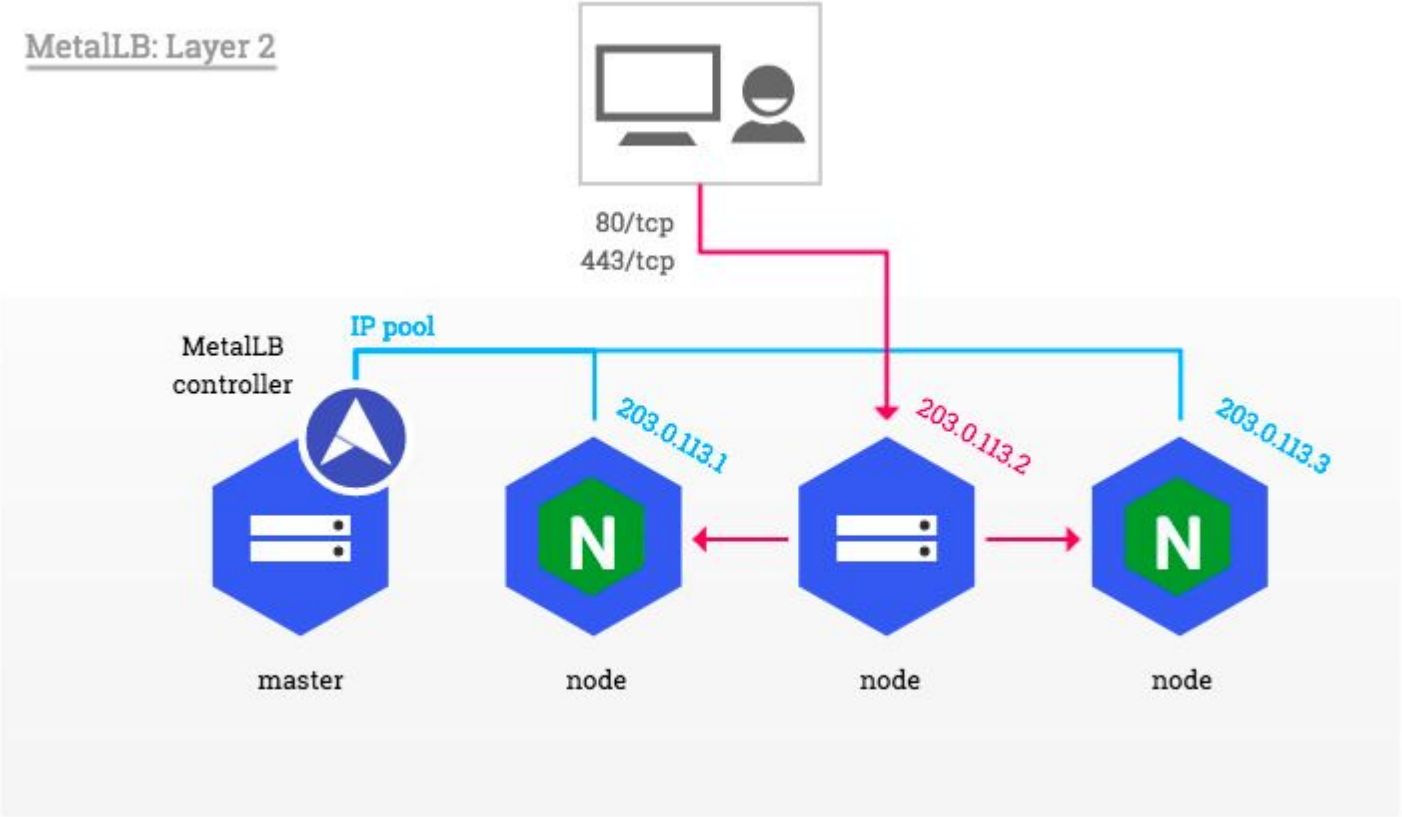
```yaml
apiVersion: v1
data:
  tls.crt: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURMVENDQWhXZ0F3SUJBZ0lK
  tls.key: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFb3dJQkFBS0NBUUVB
kind: Secret
metadata:
  name: webtest1secret
  namespace: default
type: Opaque
```

Secret: webtest2

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

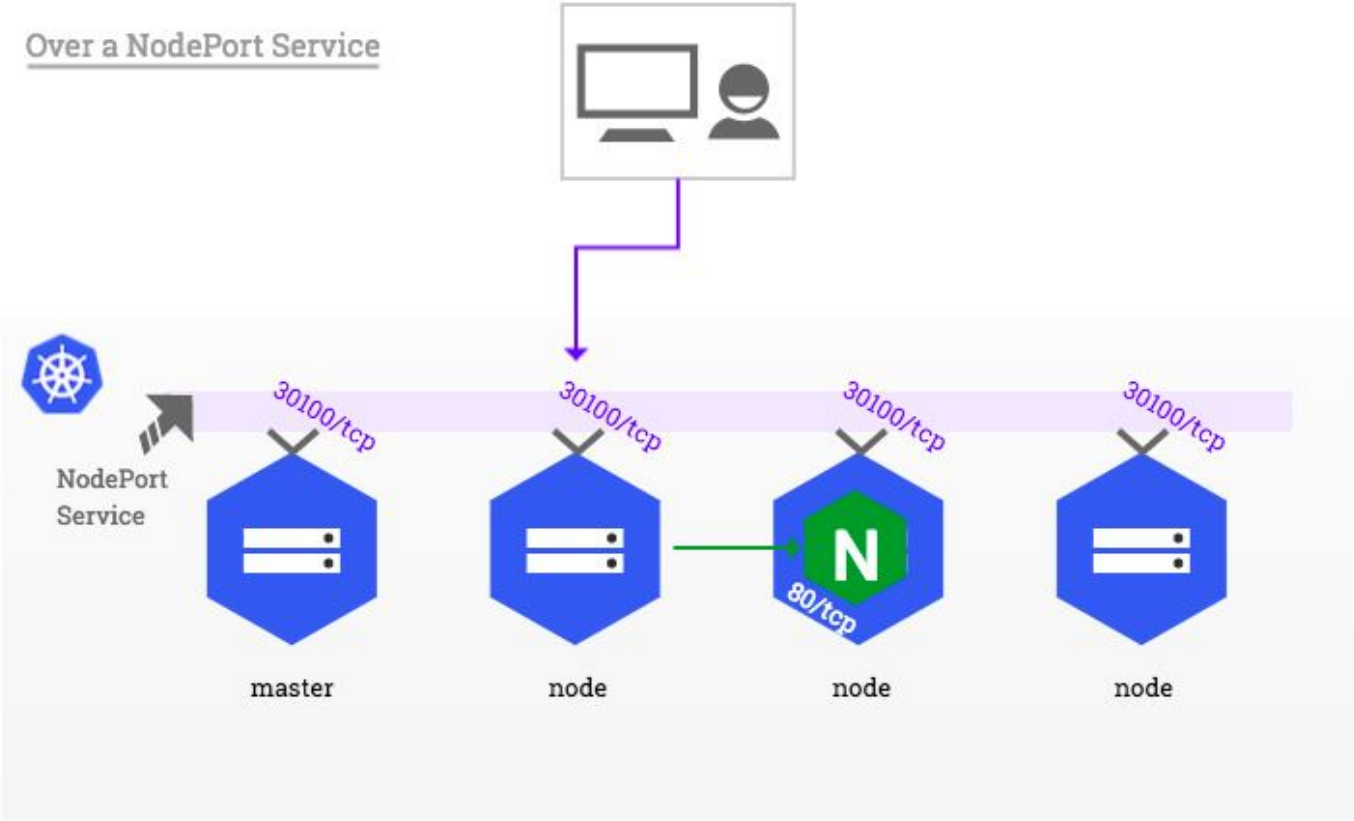
Ingress Network

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



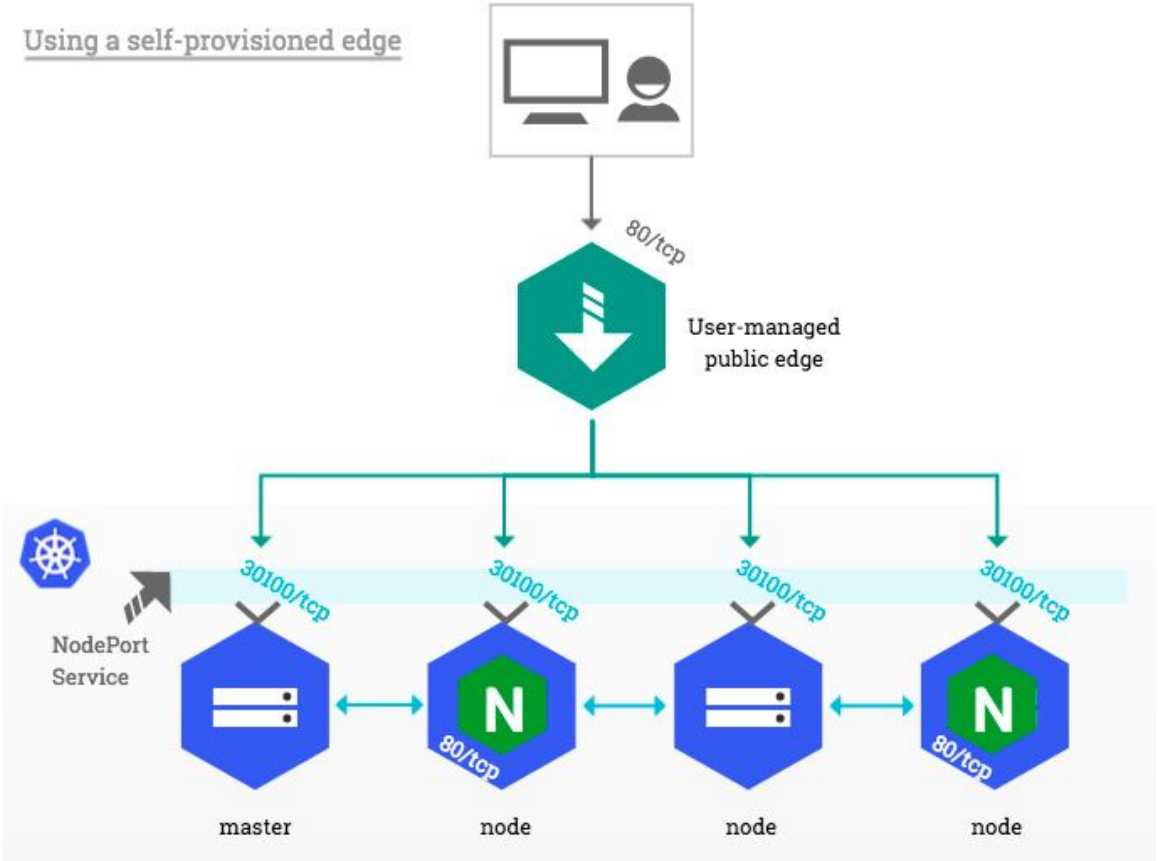
Ingress Network

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



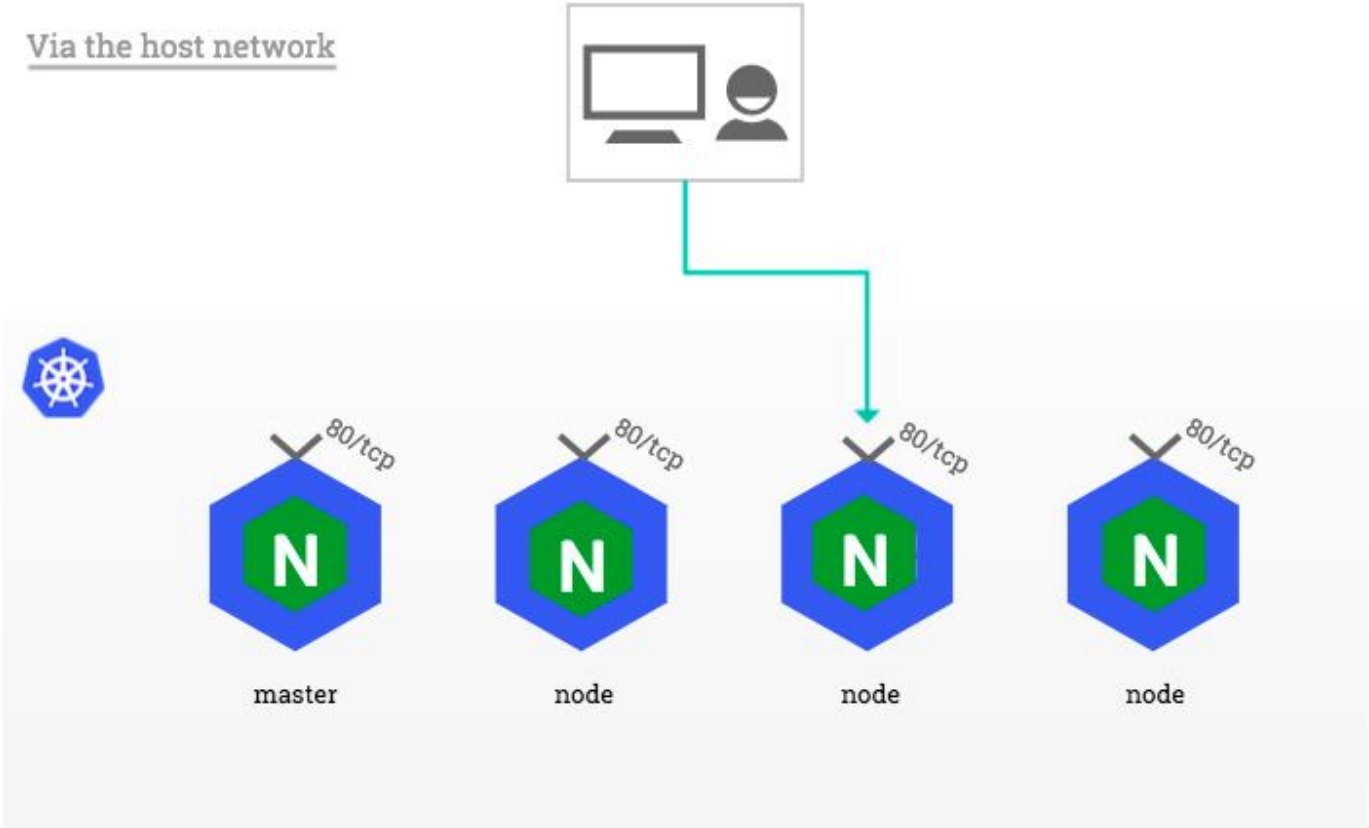
Ingress Network

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



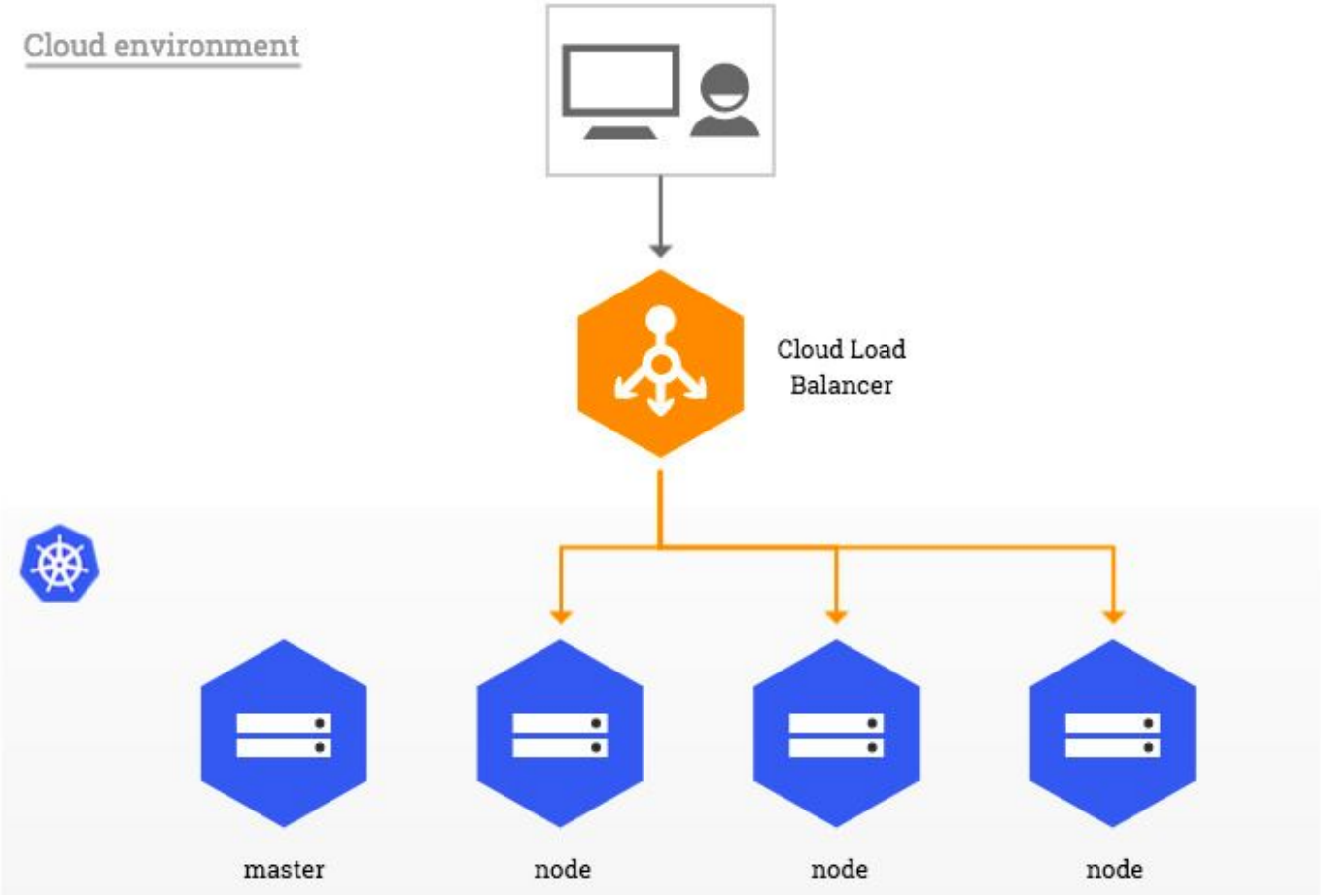
Ingress Network

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



Ingress Network

- Ingress for Cloud strategy

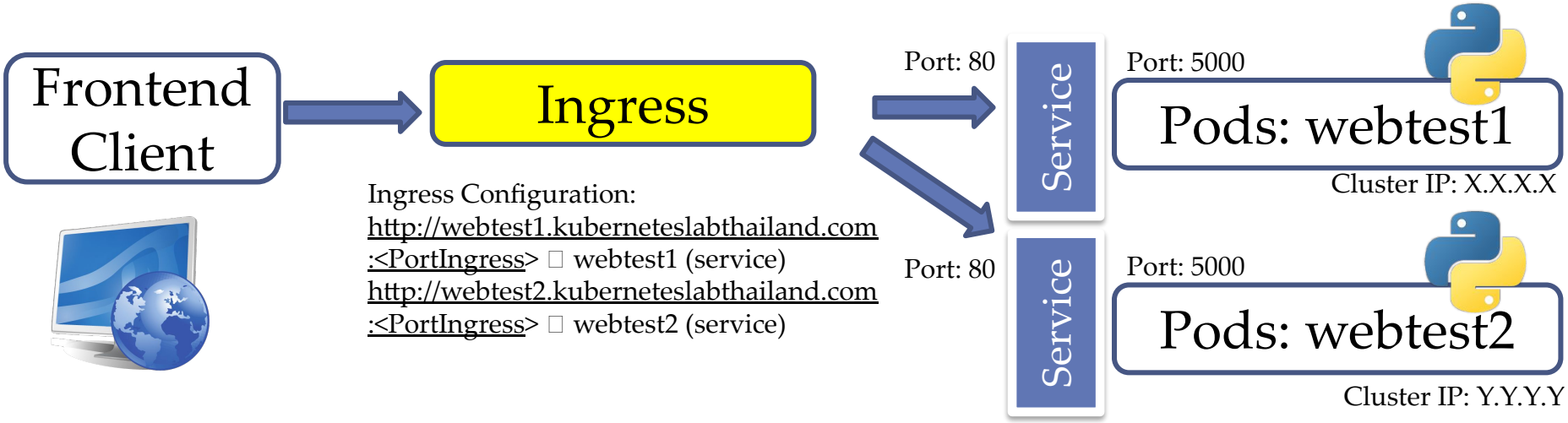


Ingress Network

- 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

Workshop: Ingress Network



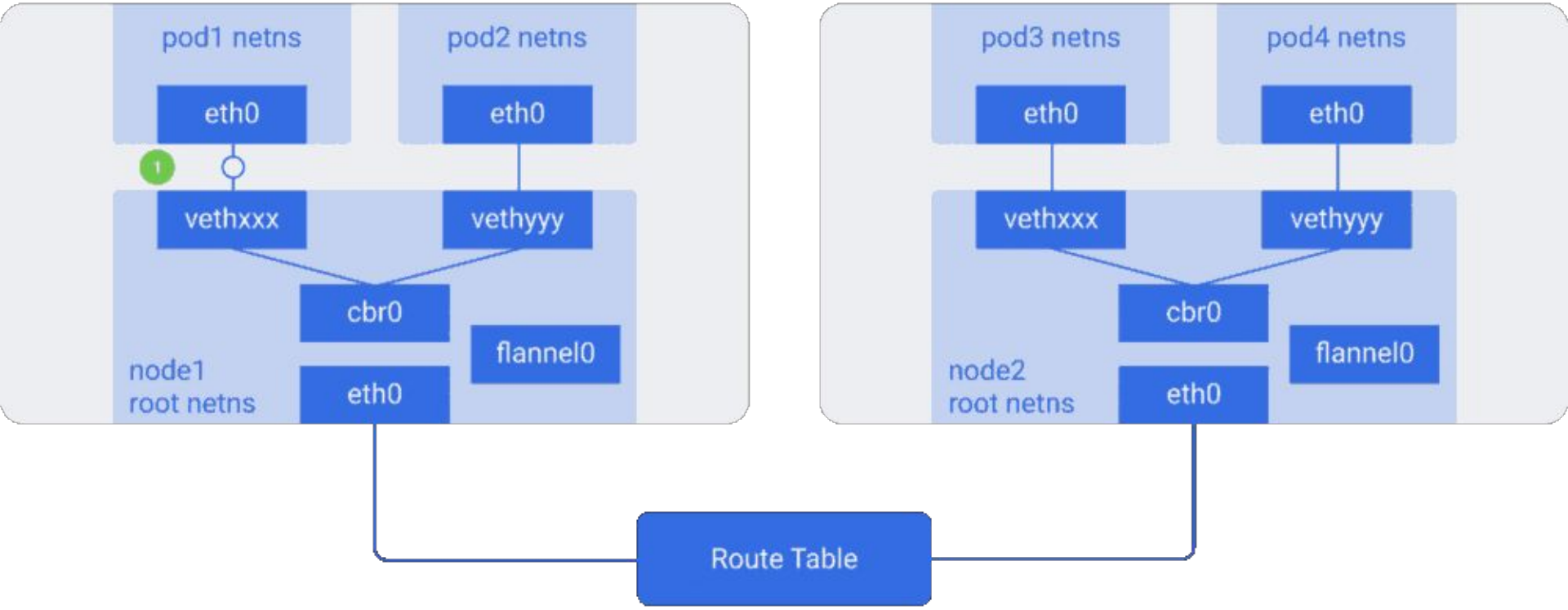
Security

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 (Alpha)**: Control security enforcemengt context (New on 1.22)

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

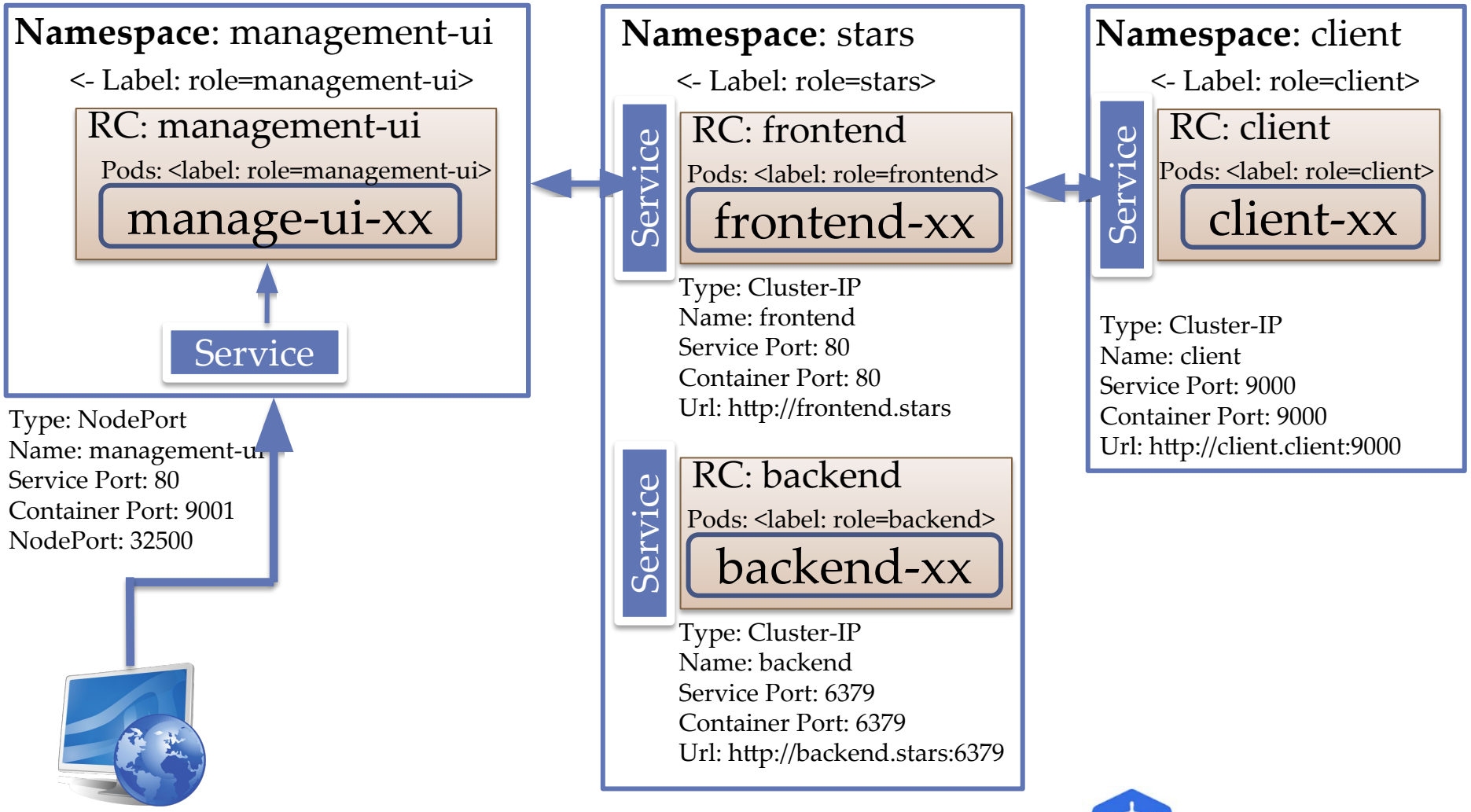
```yaml
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: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

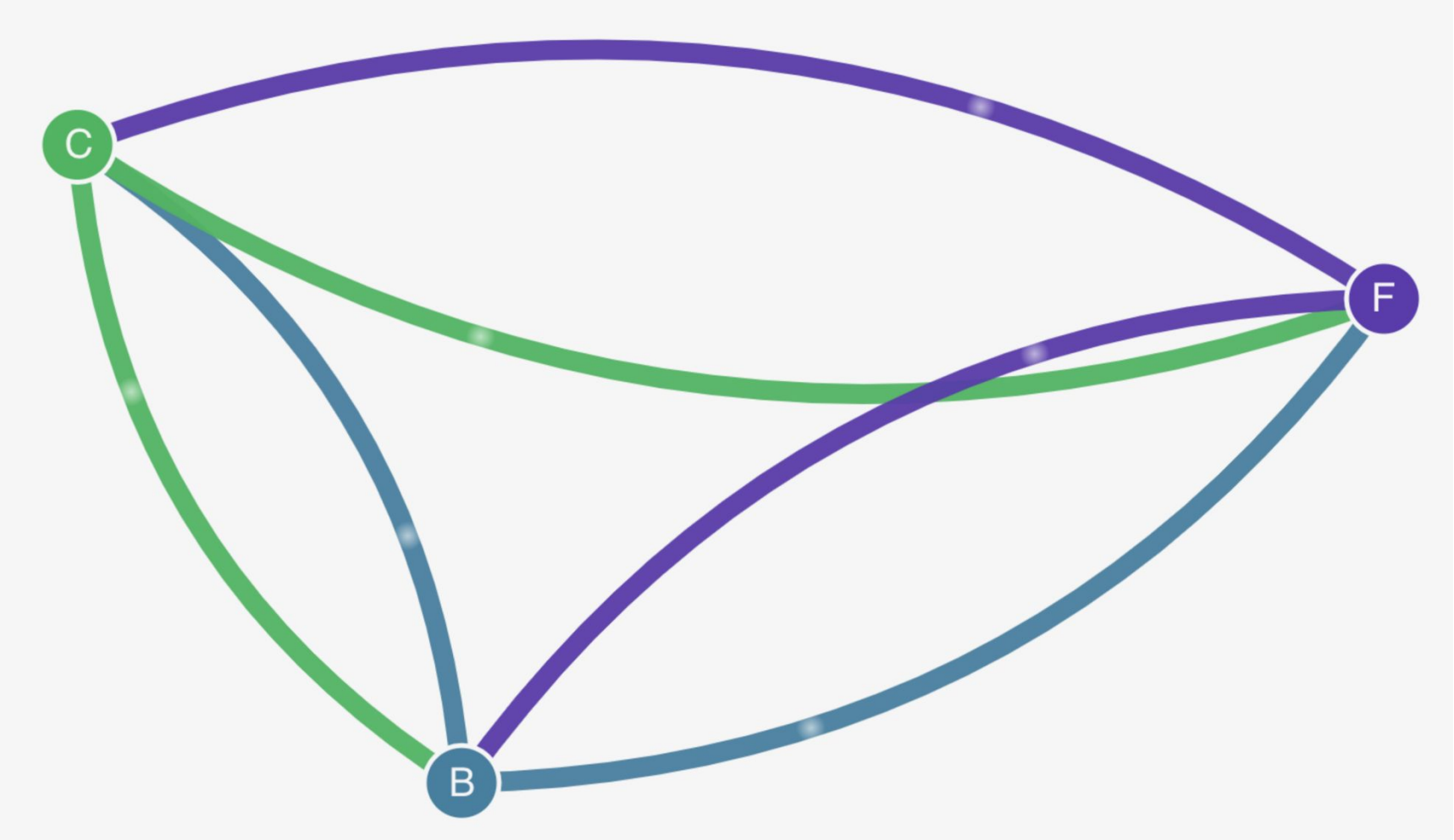
Security

- 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

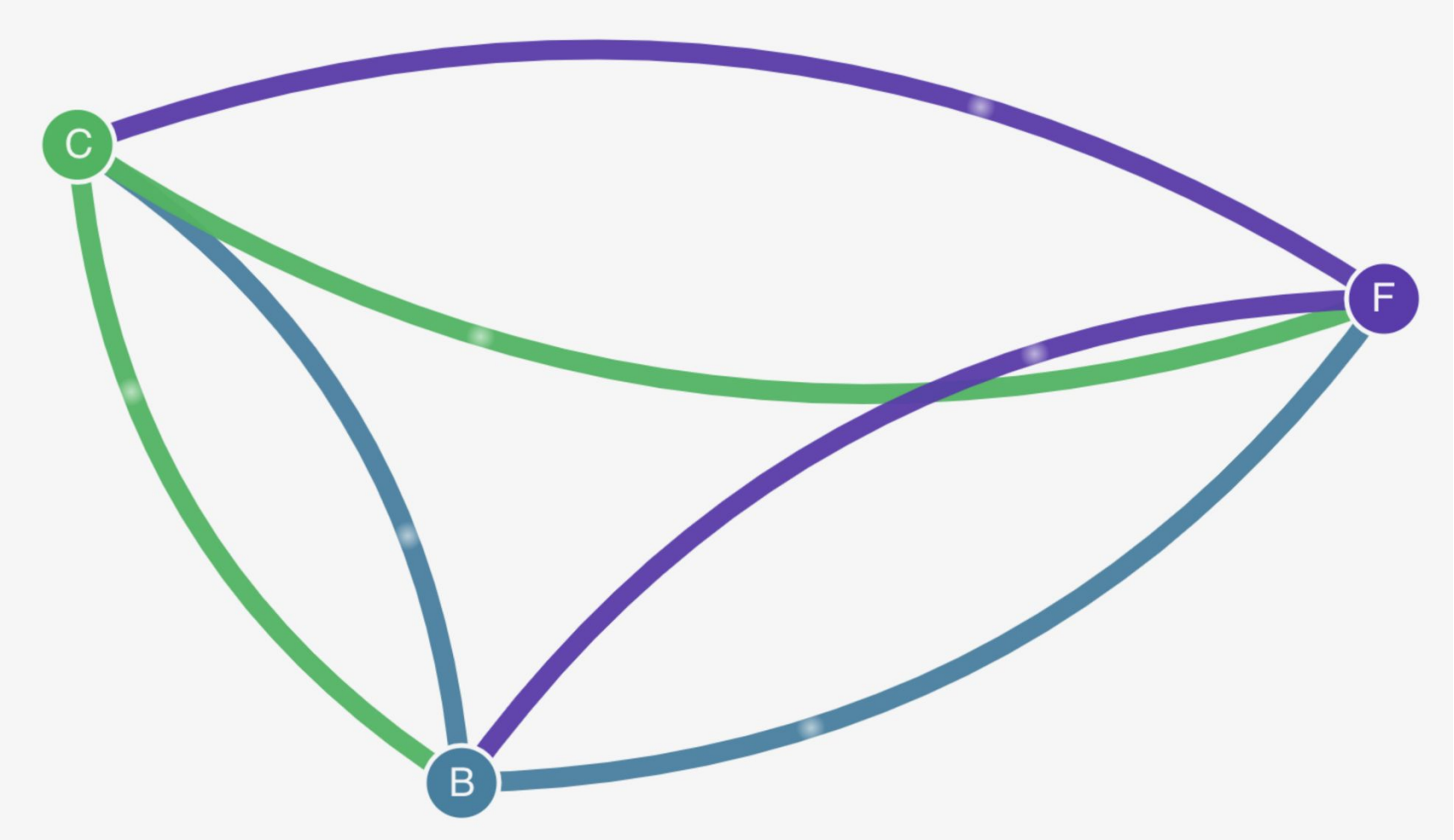
kubernetes by Google

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"

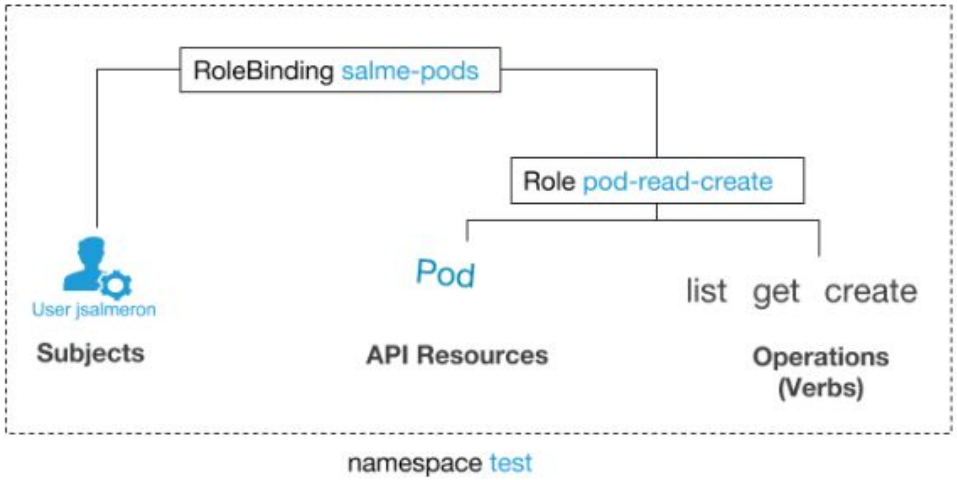
```yaml
apiVersion: v1
kind: LimitRange
metadata:
  name: webtest_limit
  labels:
    name: webtest_limit
    owner: Praparn_L
    version: "1.0"
    module: LimitRange
    environment: development
spec:
  limits:
  - max:
      cpu: "1"
      memory: 1Gi
    min:
      cpu: 200m
      memory: 6Mi
 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

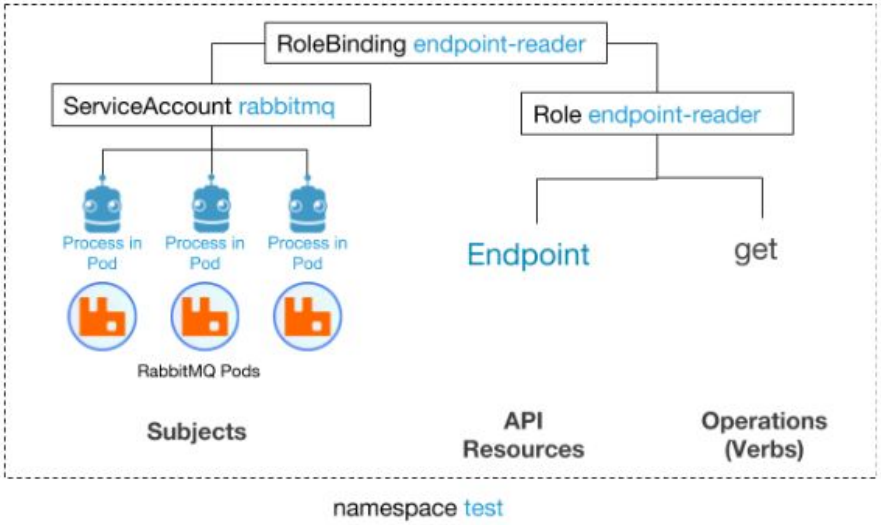
Security

- 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/

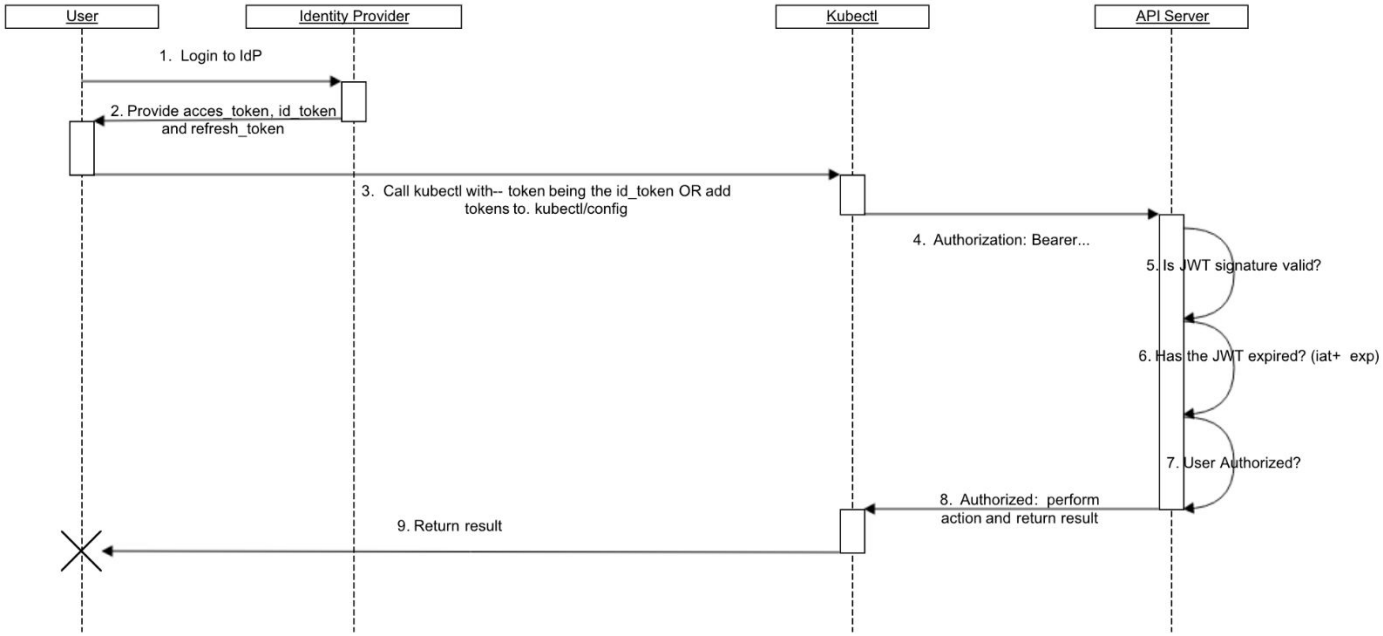
Security

- 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

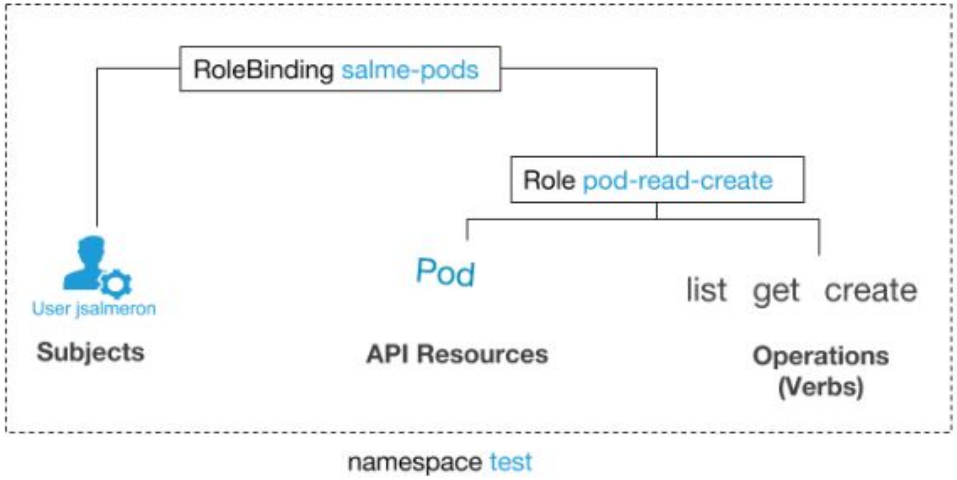
- Access Control Policy:
  - **RBAC Operate**
  - Aggregated ClusterRoles
  - Combine of multiple cluster roles
    - Match by label and rule will automatic filled

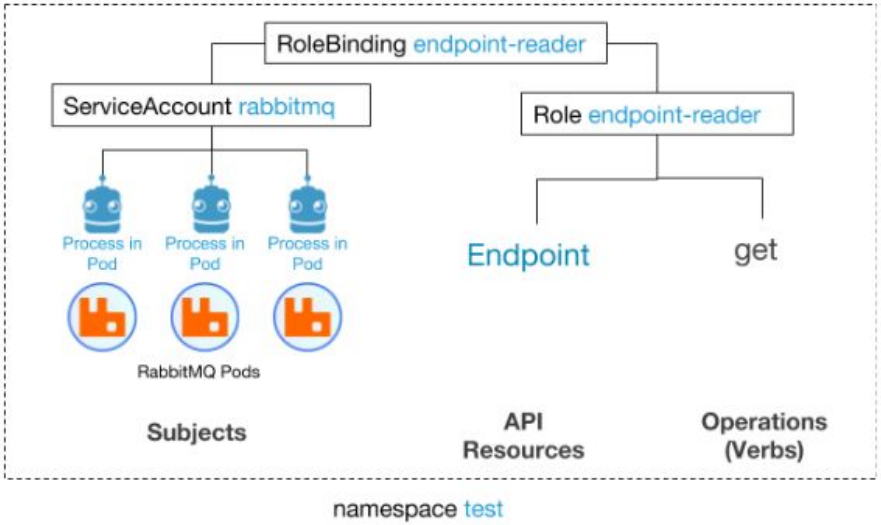
```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: templateaggregateclusterrole
  namespace: security
aggregationRule:
  clusterRoleSelectors:
  - matchLabels:
      rbac.kubernetesthailand.com/security: "true"
rules: [] # Rules are automatically filled in by the controller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

```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`

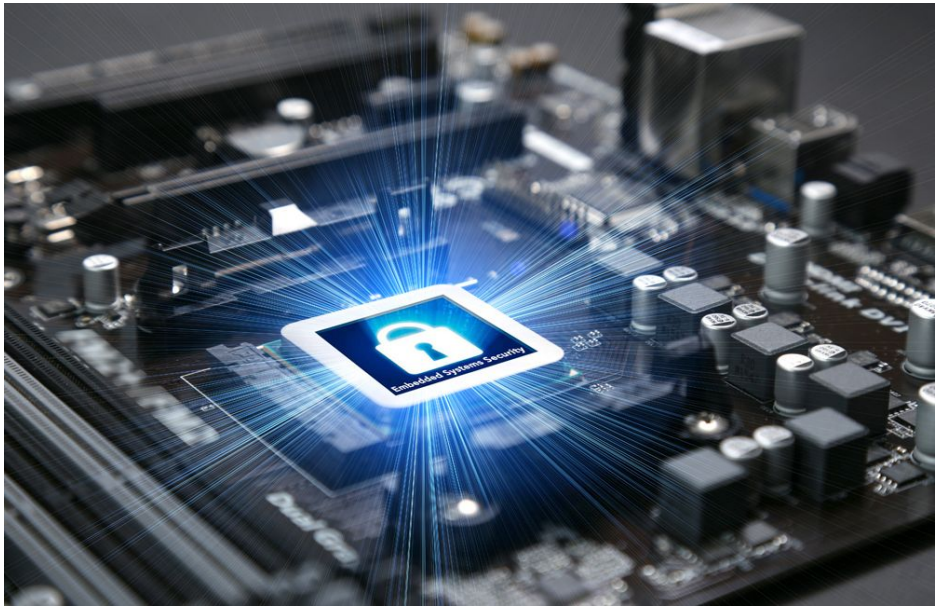
Security

- 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: v1.23

    # We are setting these to our _desired_ `enforce` level.
    pod-security.kubernetes.io/audit: restricted
    pod-security.kubernetes.io/audit-version: v1.23
    pod-security.kubernetes.io/warn: restricted
    pod-security.kubernetes.io/warn-version: v1.23
```

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

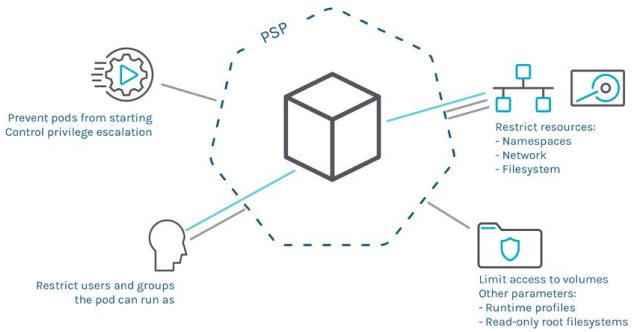
- Example: Pods security context

```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

Recap Day 4

- Fundamental of Kubernetes
  - Configmap and Secret
  - 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

Q&A

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

DAY 5

Outline Day 5

- 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

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

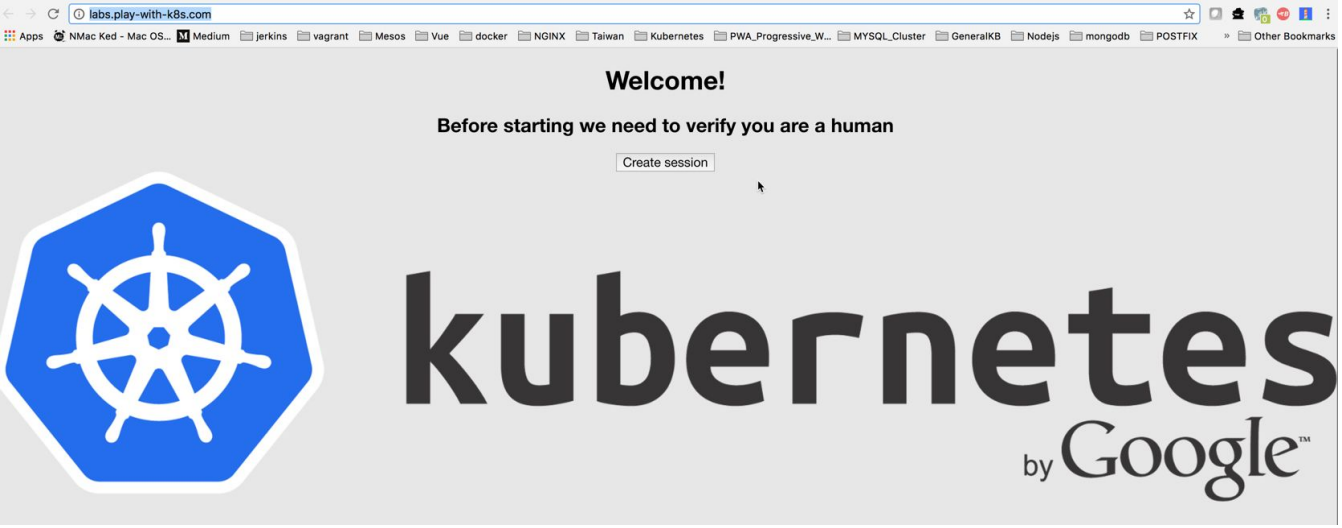
To install kubeadm, see the installation guide.

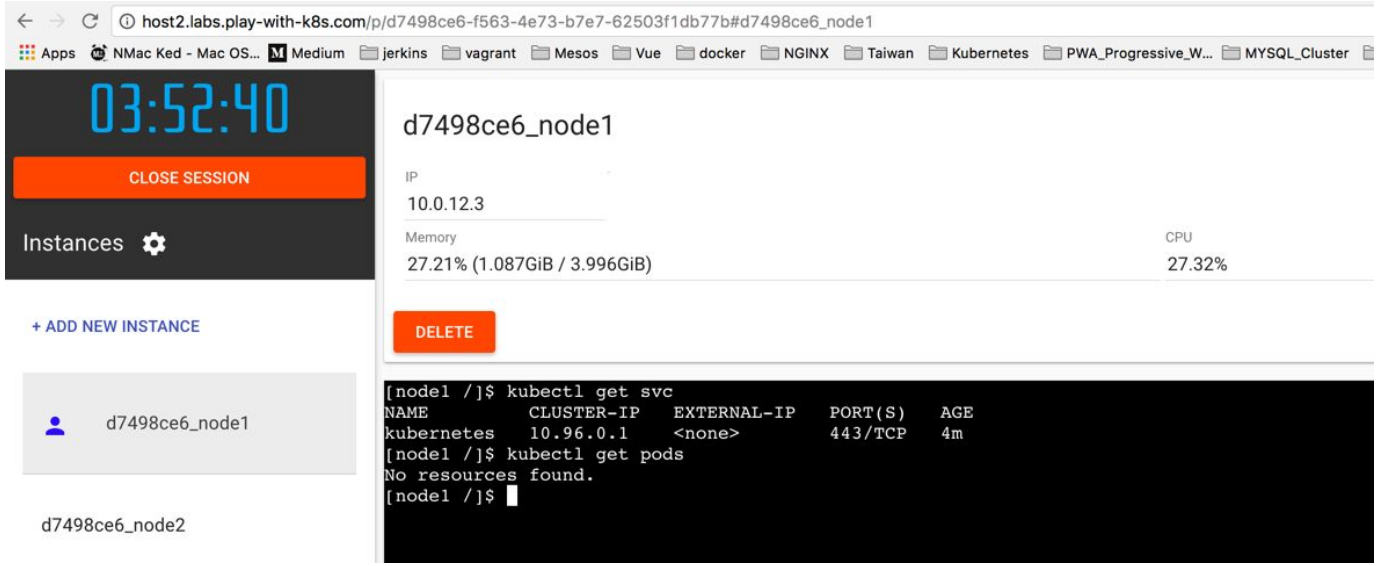
What's next

- 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



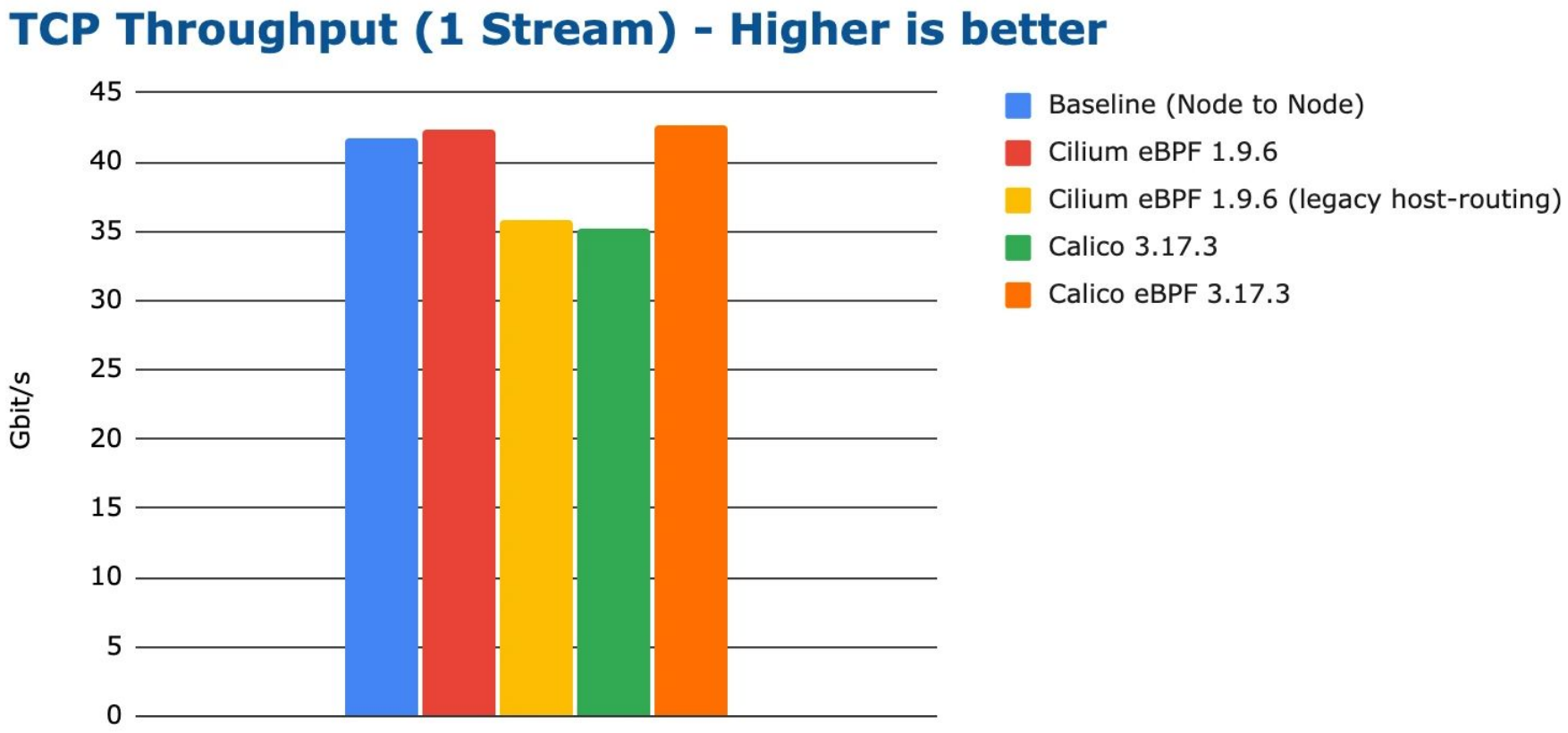


Kubernetes: Production Workload Orchestration

kubernetes by Google

Kubernetes in real world

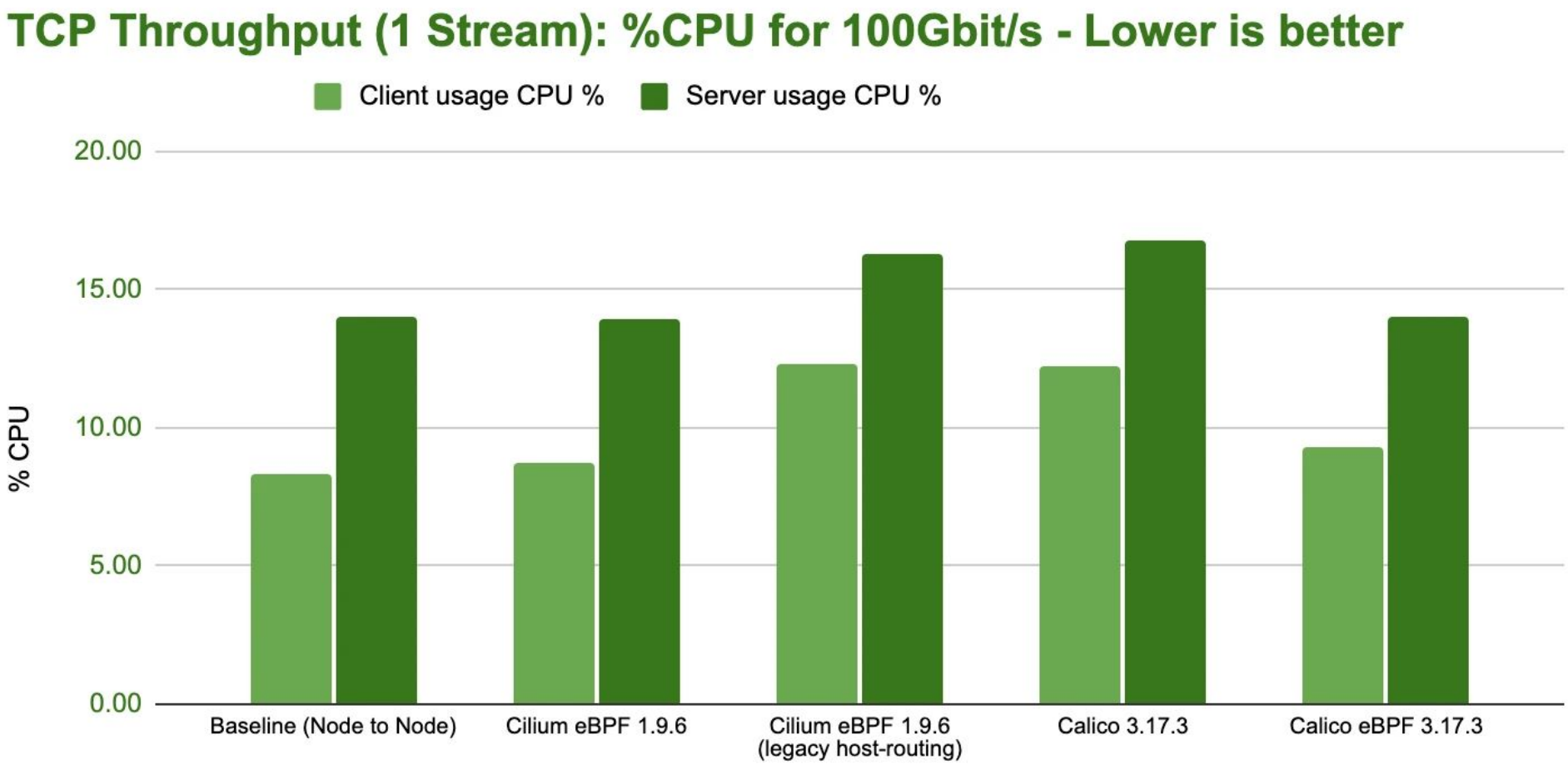
- Network Optional



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

Kubernetes in real world

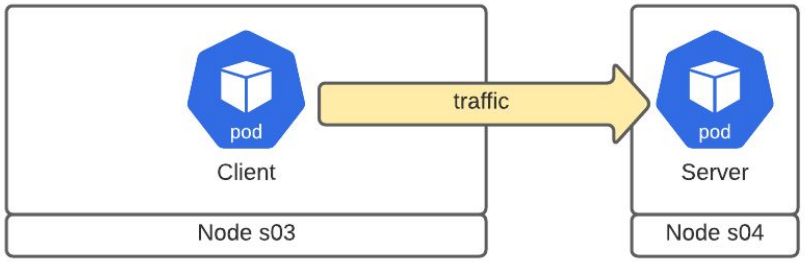
- Network Optional

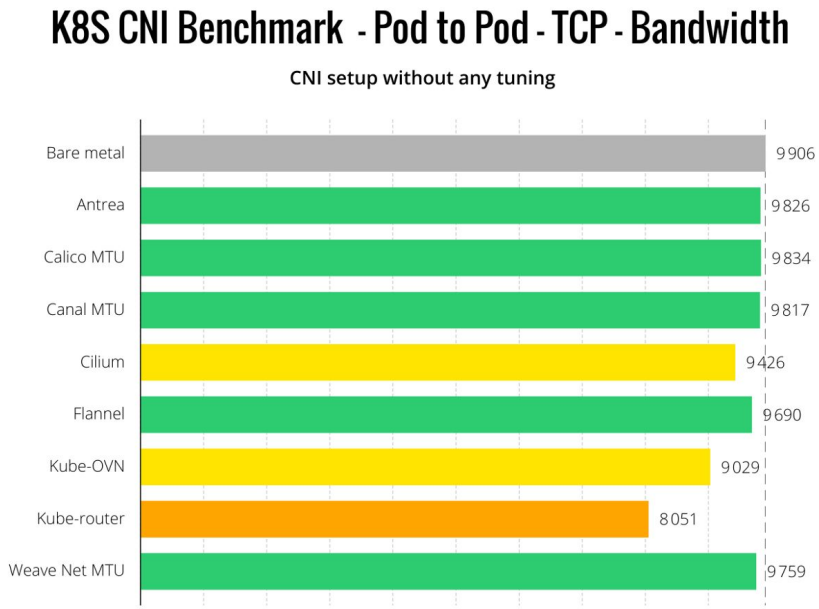


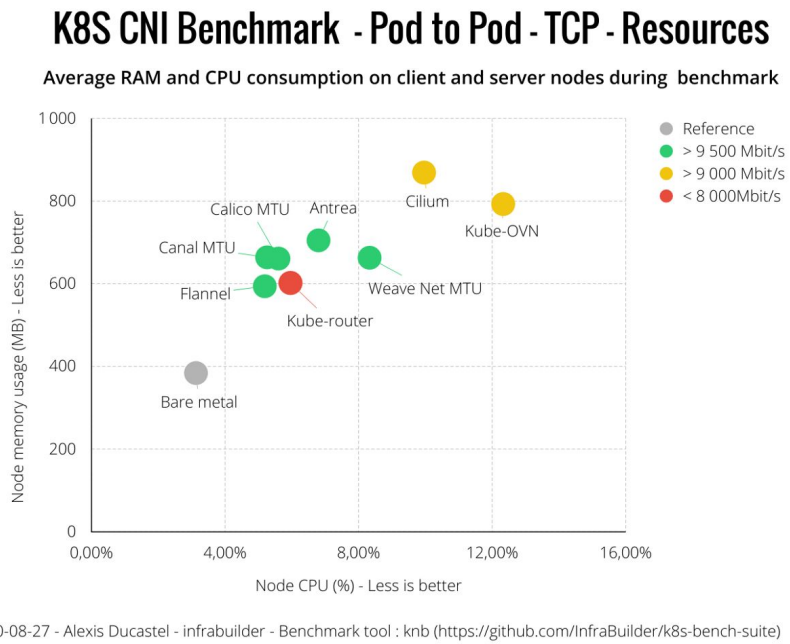
Ref: https://cilium.io/blog/2021/05/11/cni-benchmark

Kubernetes in real world

- Network Optional



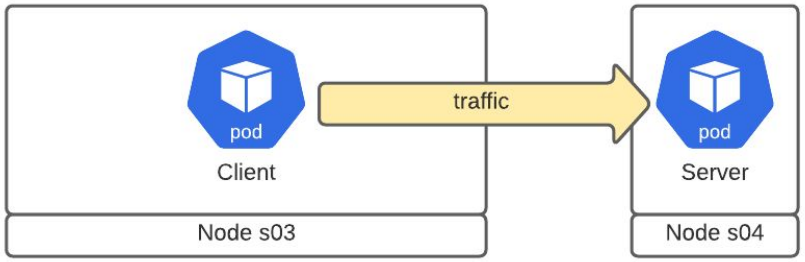


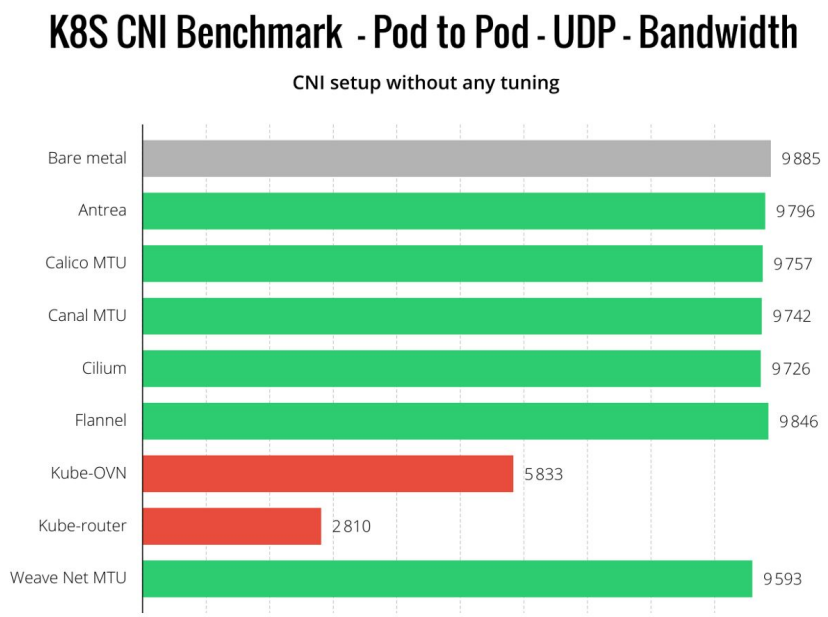


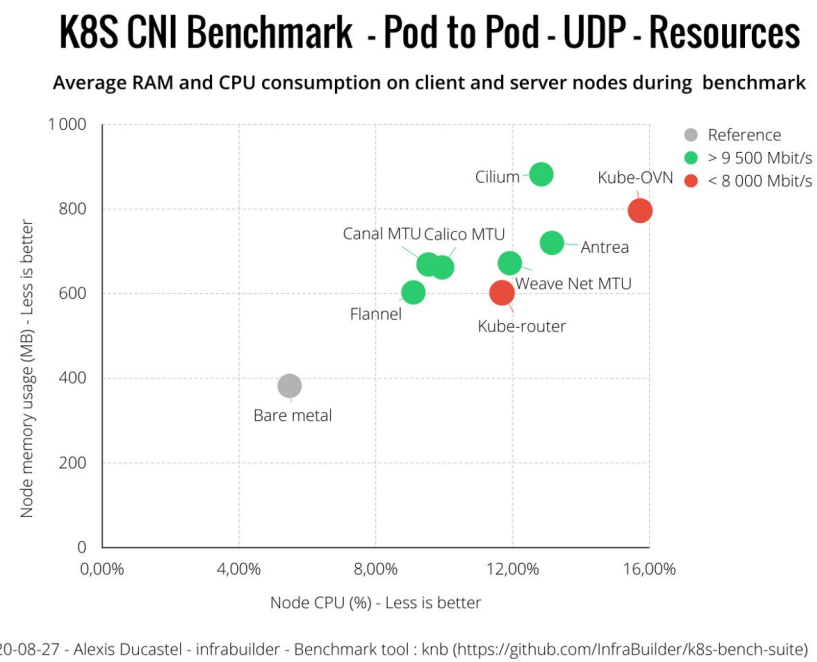
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Network Optional



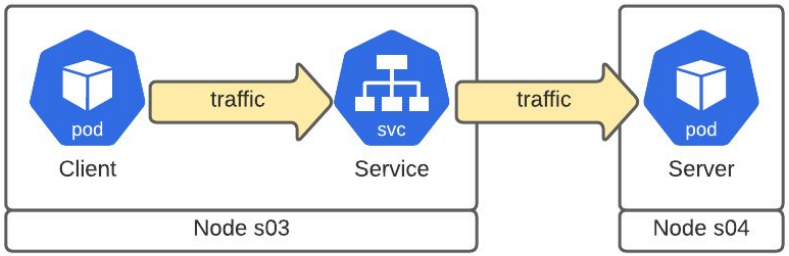


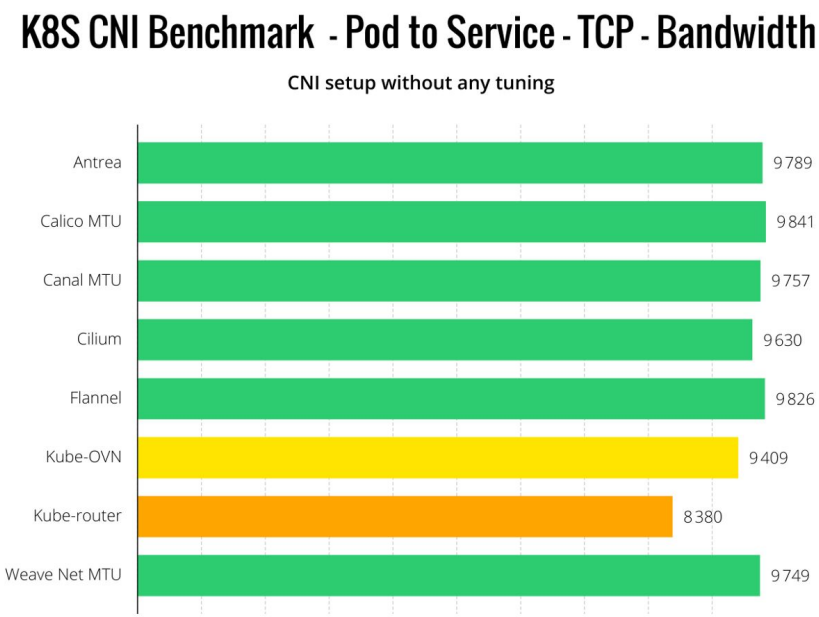


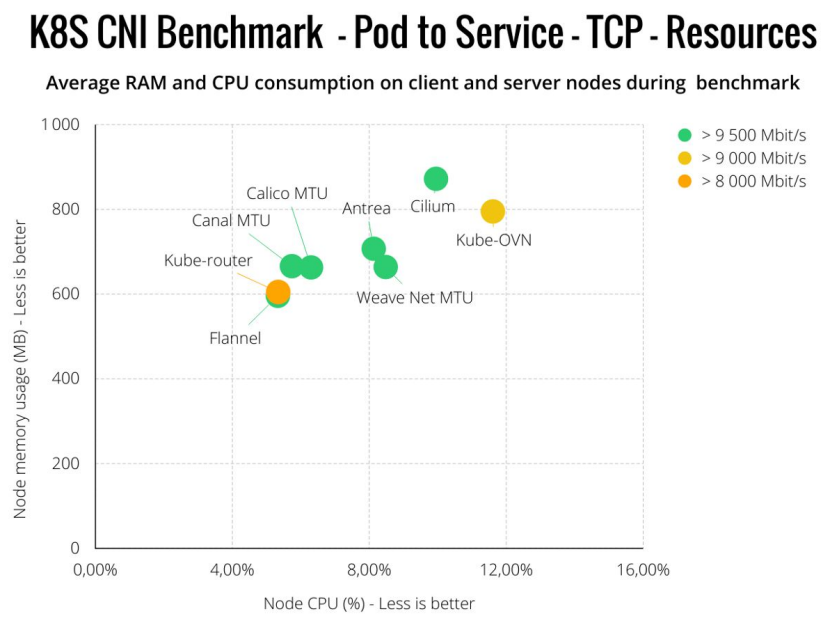
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Network Optional



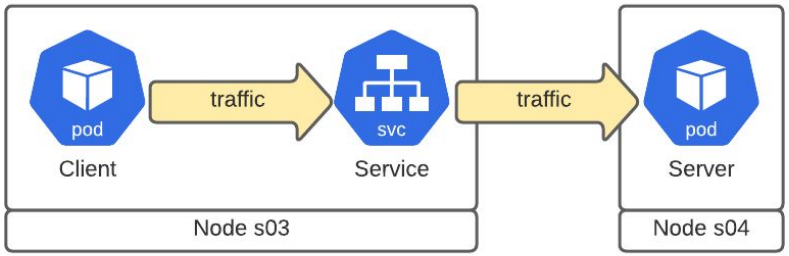


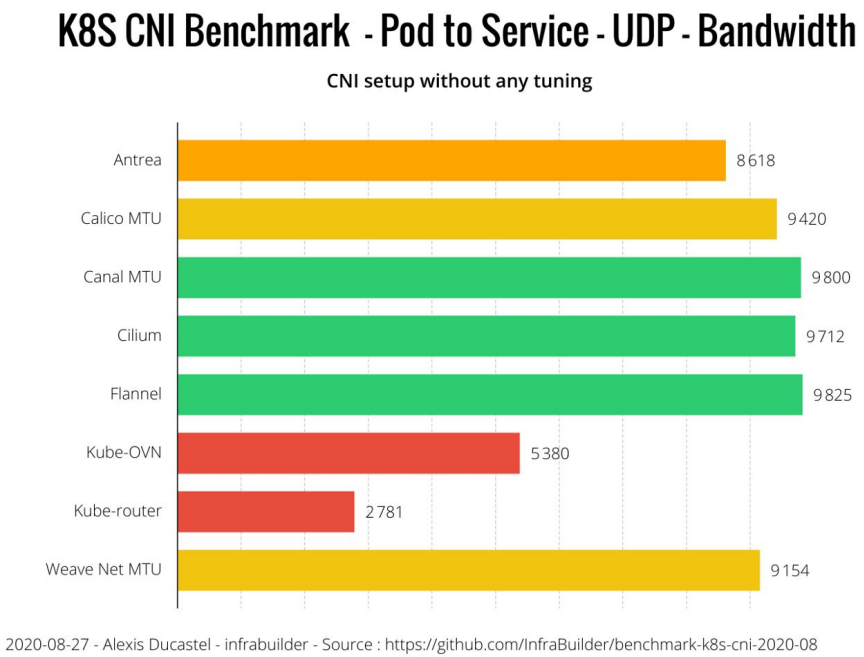


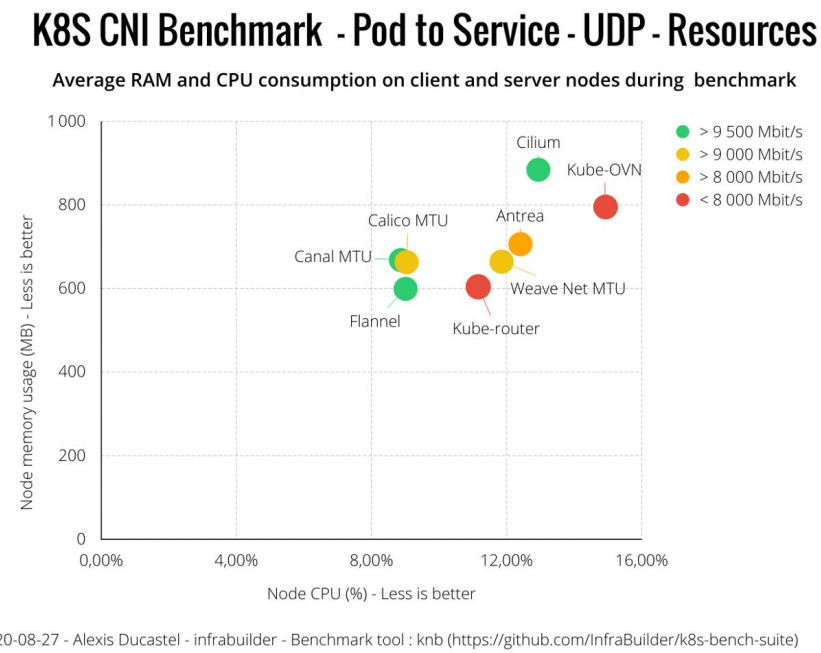
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

- Network Optional



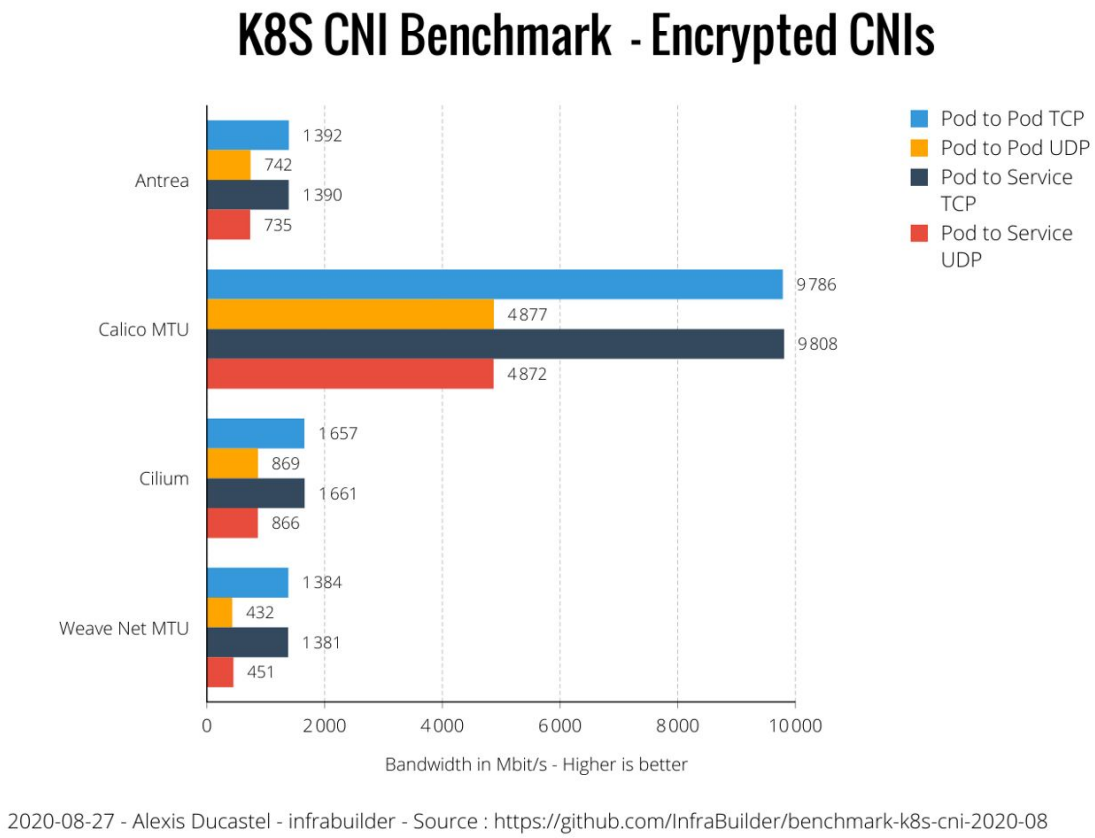




https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

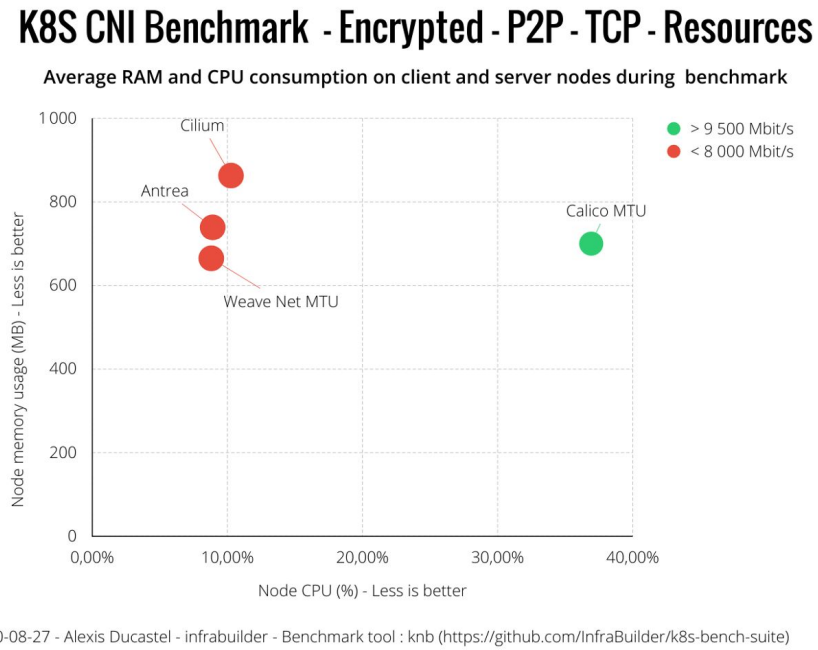
- Network Optional

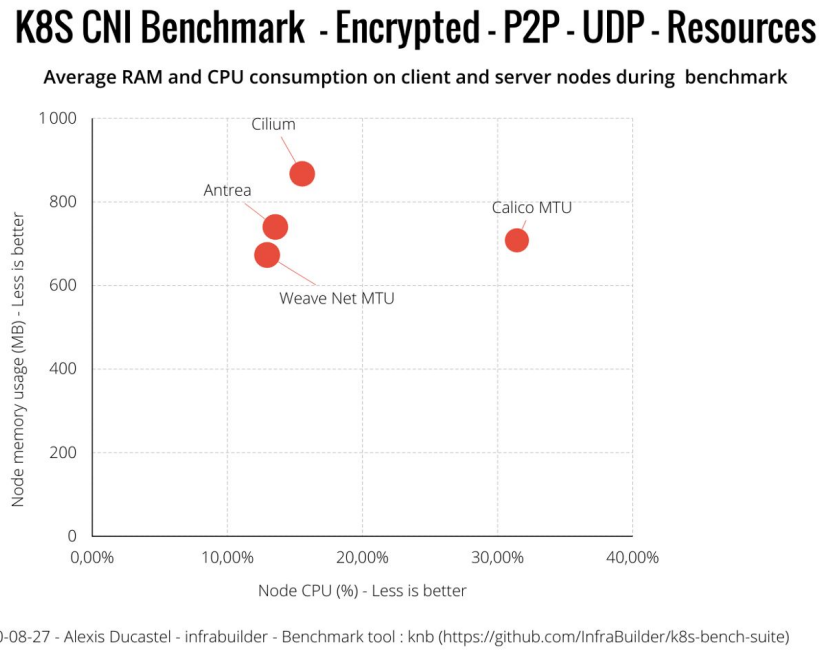


https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

Kubernetes in real world

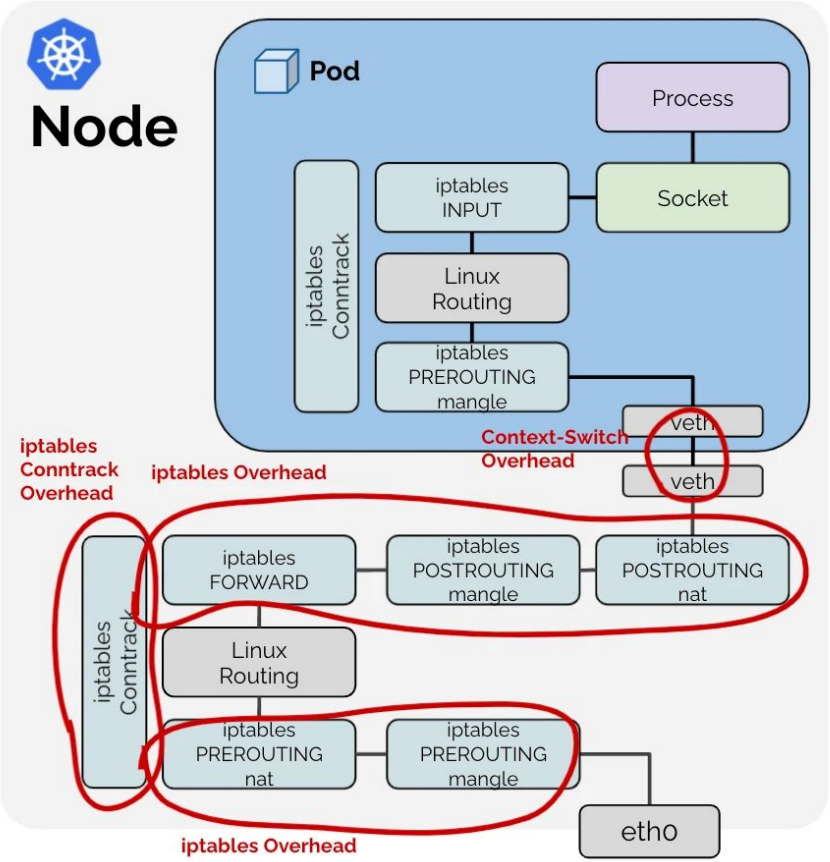
- Network Optional



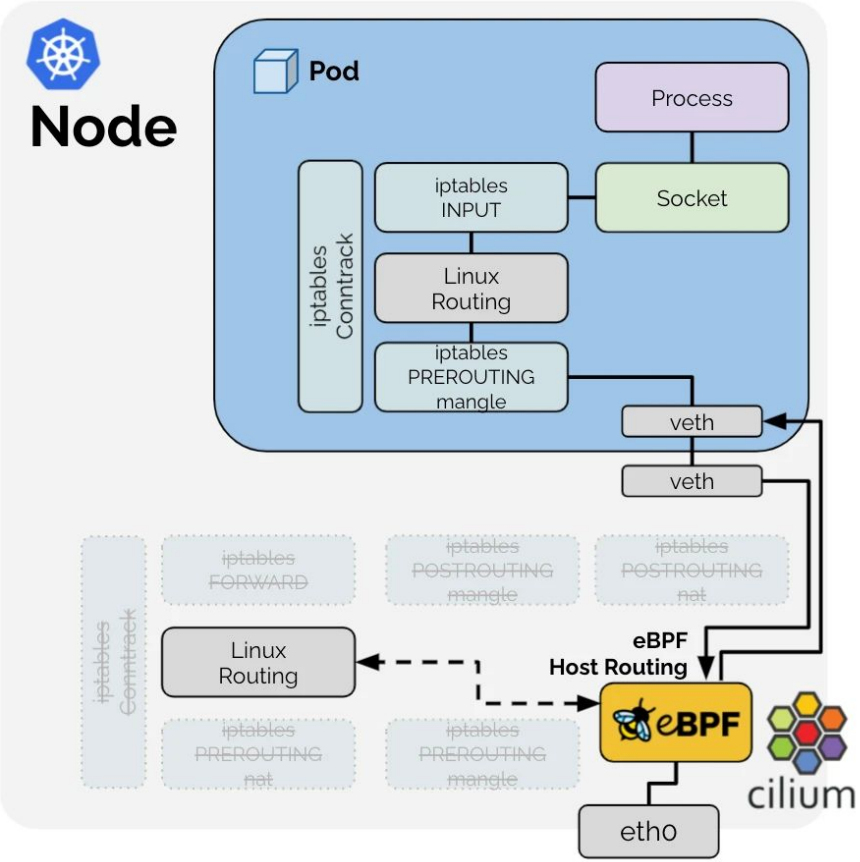


https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-august-2020-6e1b757b9e49

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

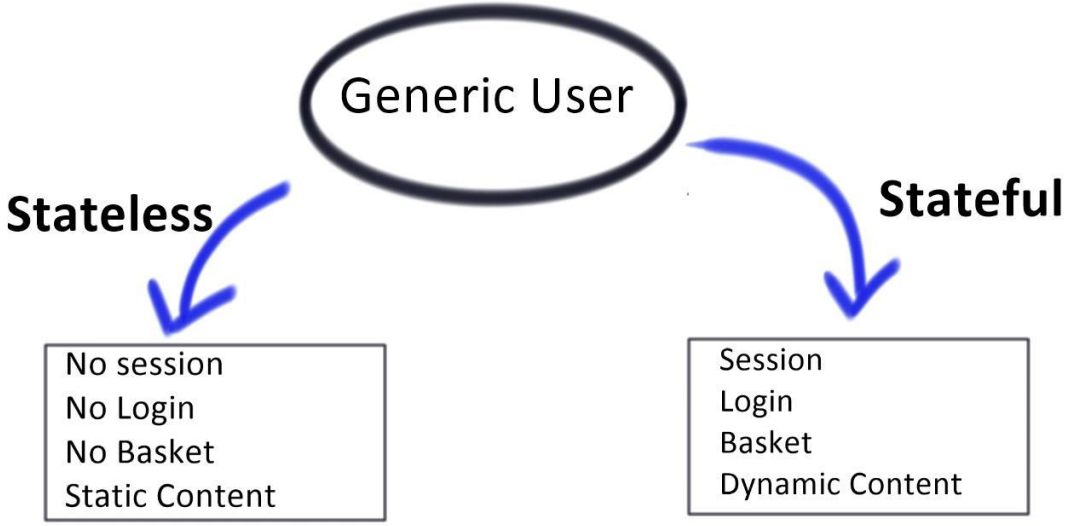
Role: Client

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

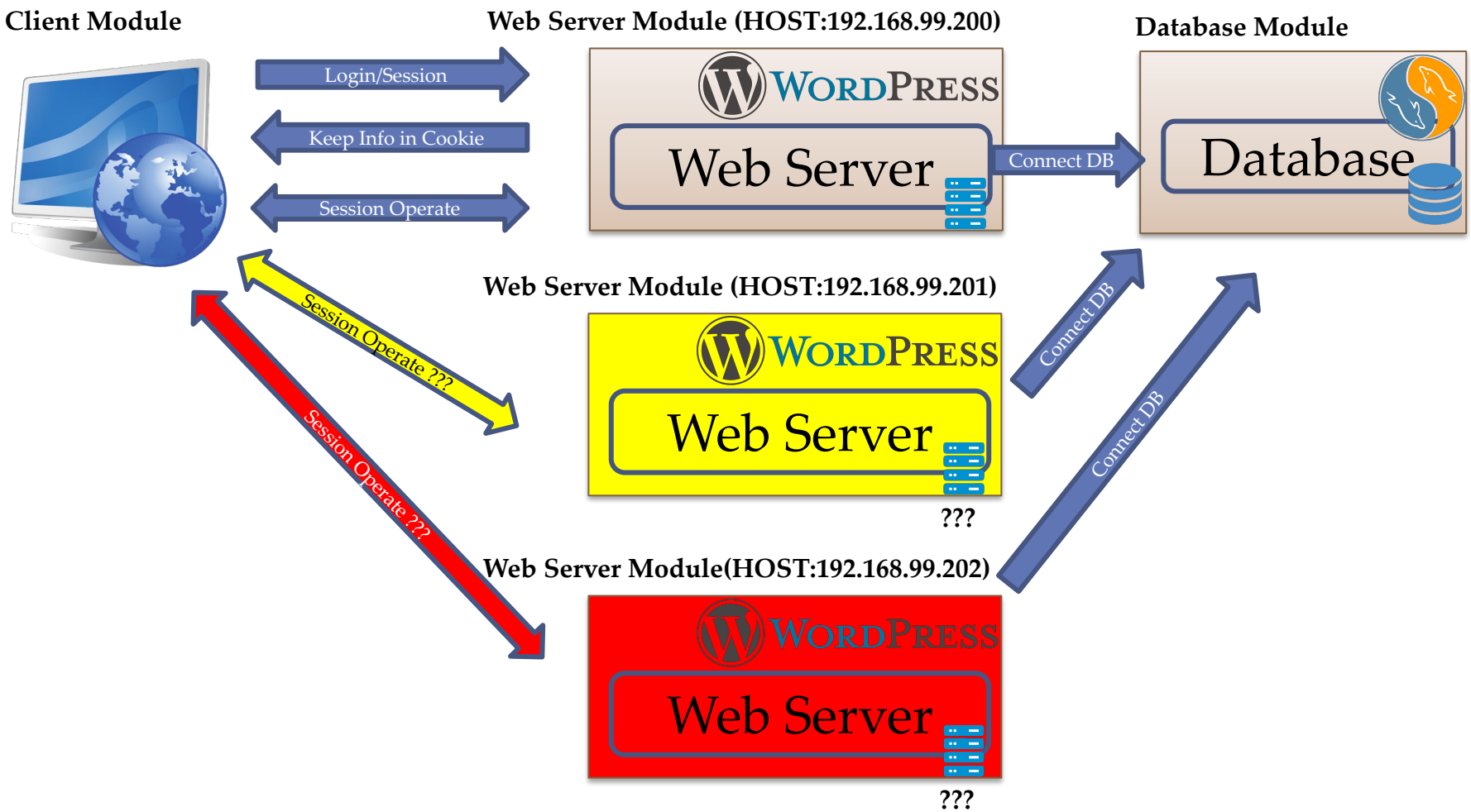


Stateful Application Deployment

- 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

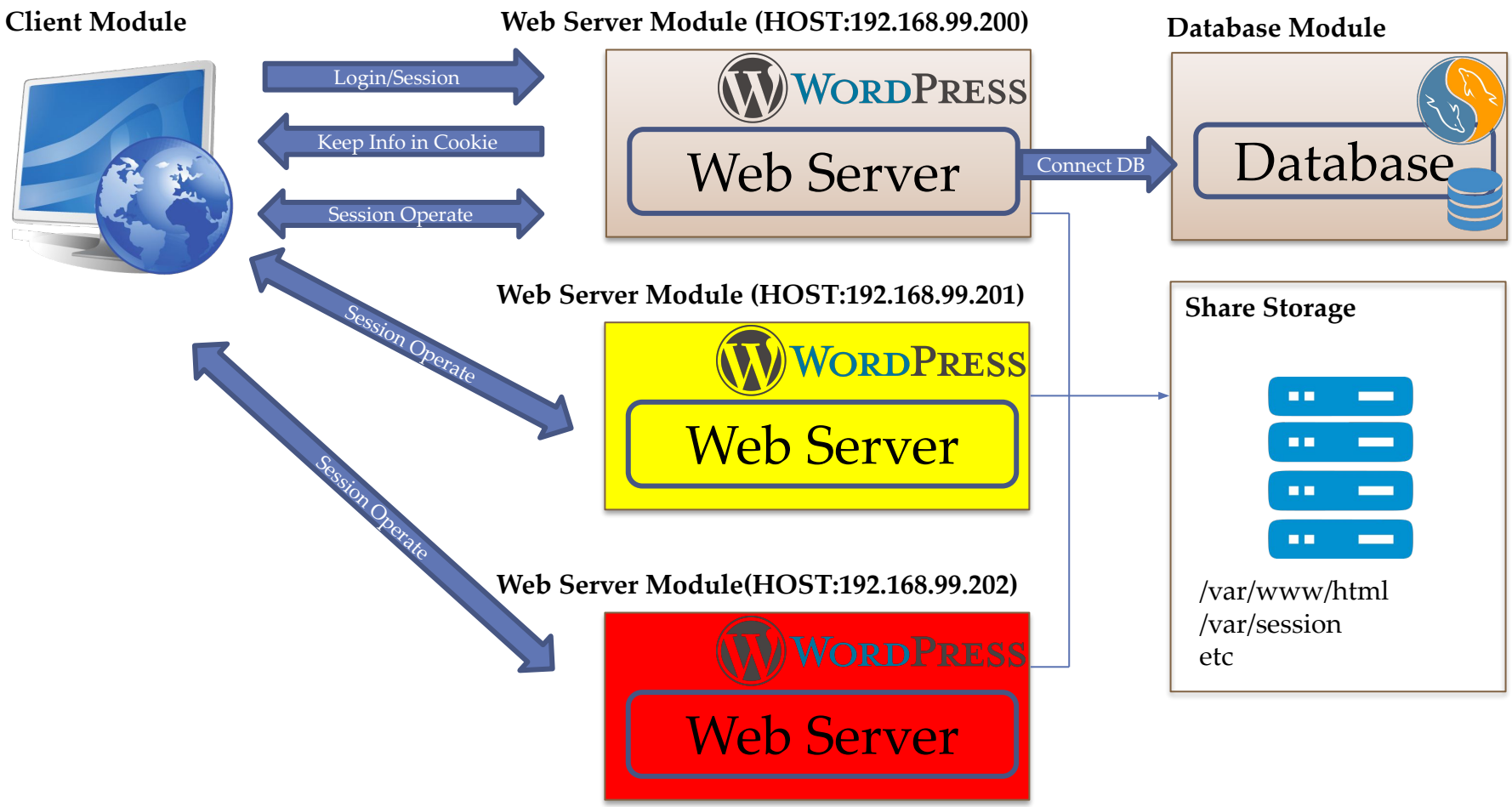
Stateful Application Deployment



Stateful Application Deployment

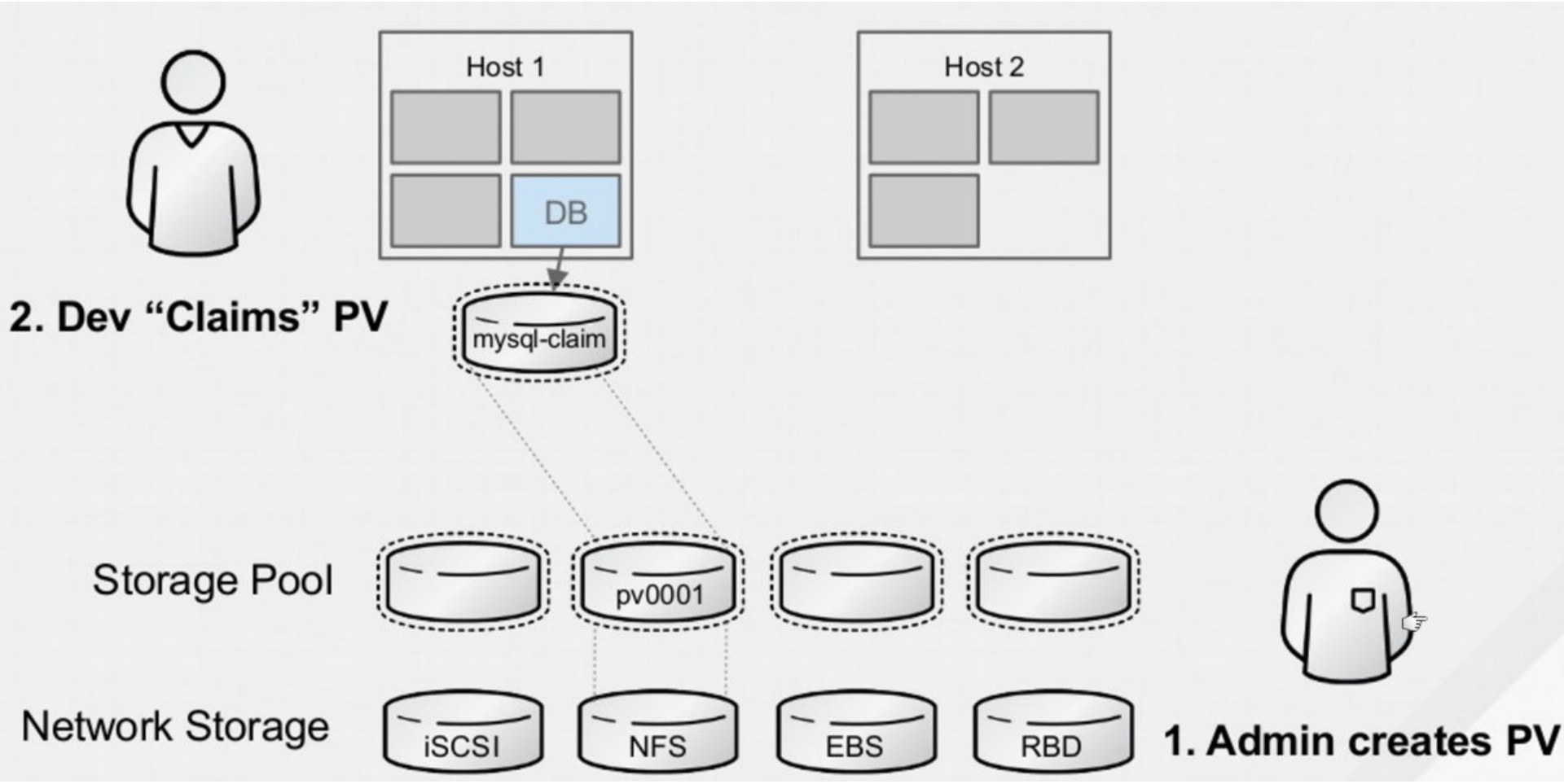
- Solution ?
  - SDS (Software-Defined Storage) for make centralize storage pool
  - Share centralize storage pool for all node
  - For Web/App Server
    - Keep application path / Session path on storage pool
    - Every server will read/write on same place
  - For Database Server
    - Many option 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



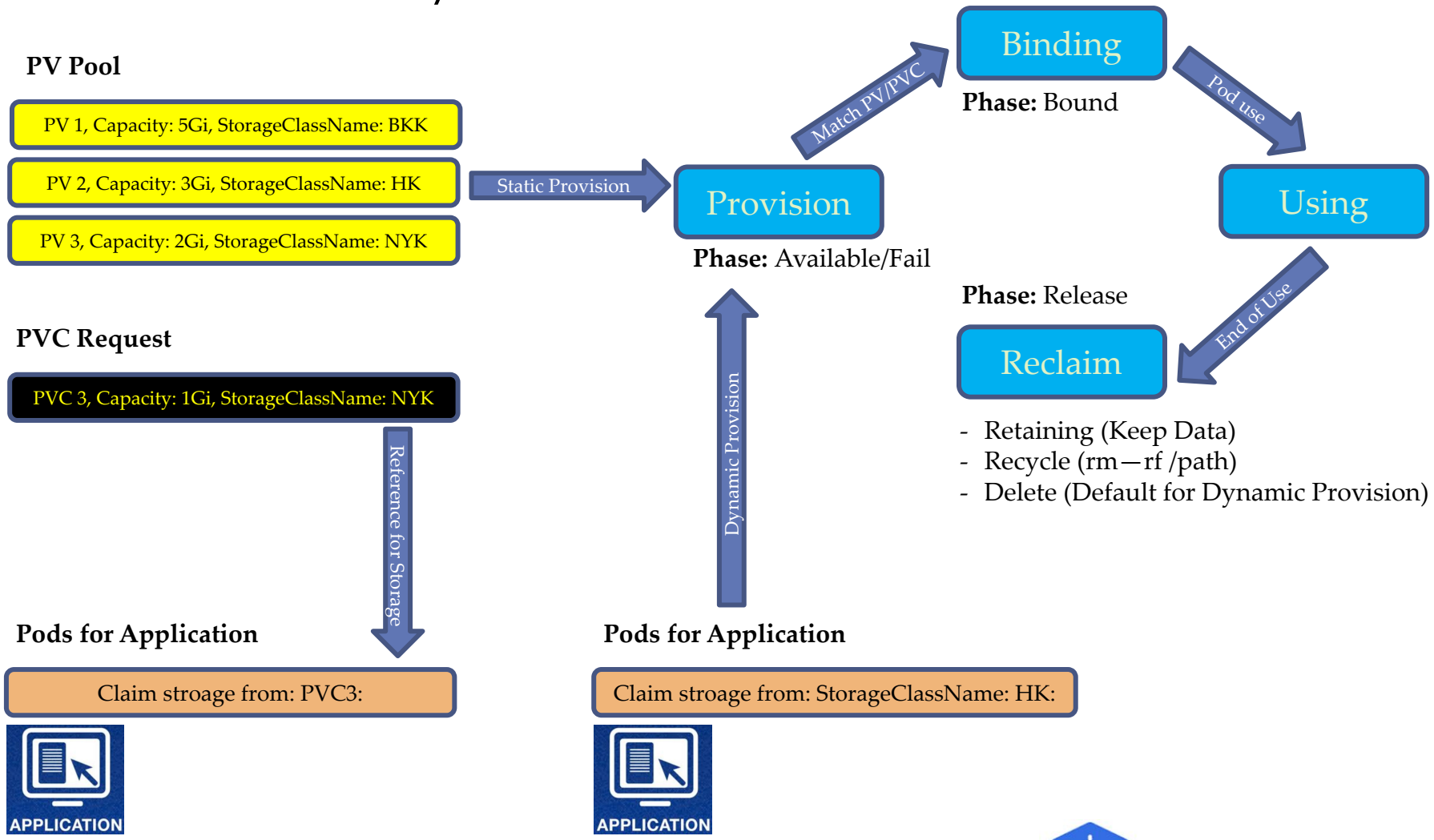
Persistent Volume

- 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)

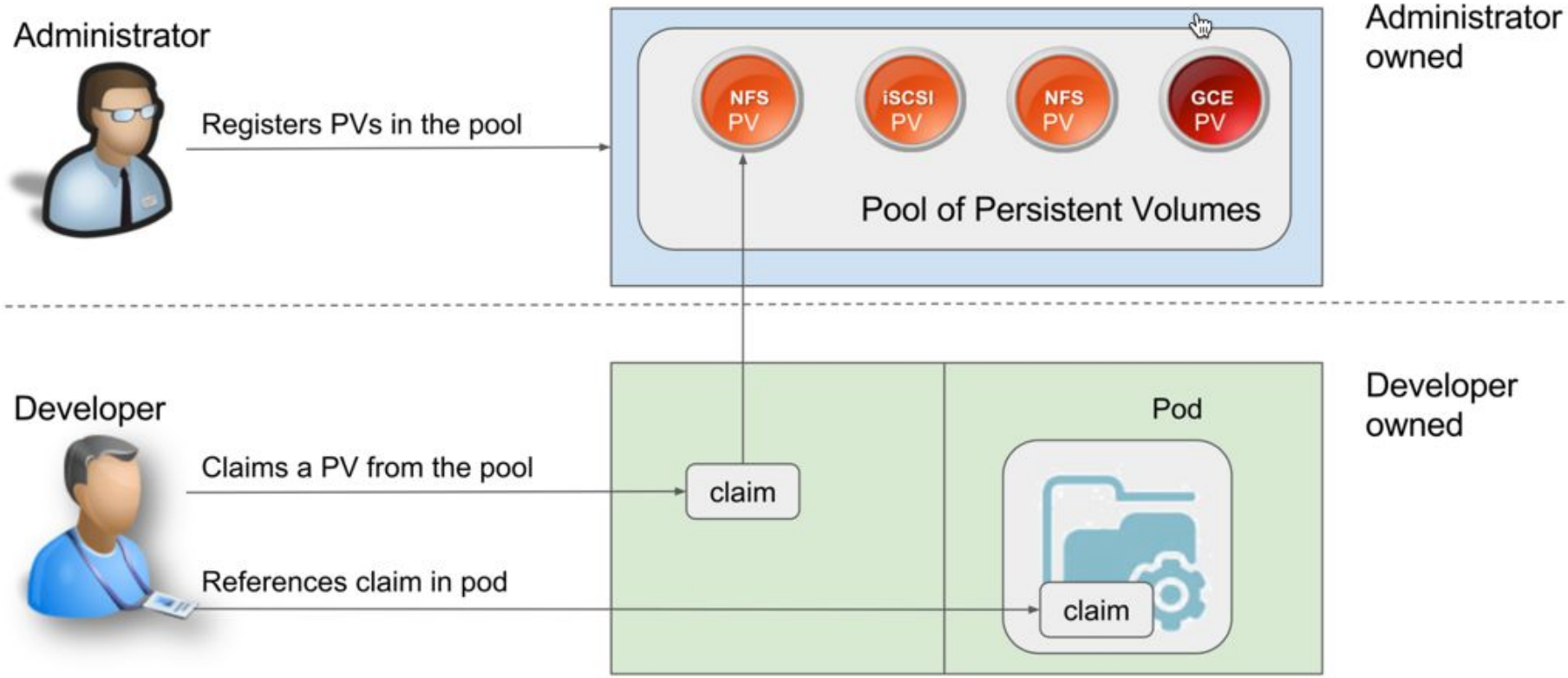
Persistent Volume

- Volume Life Cycle



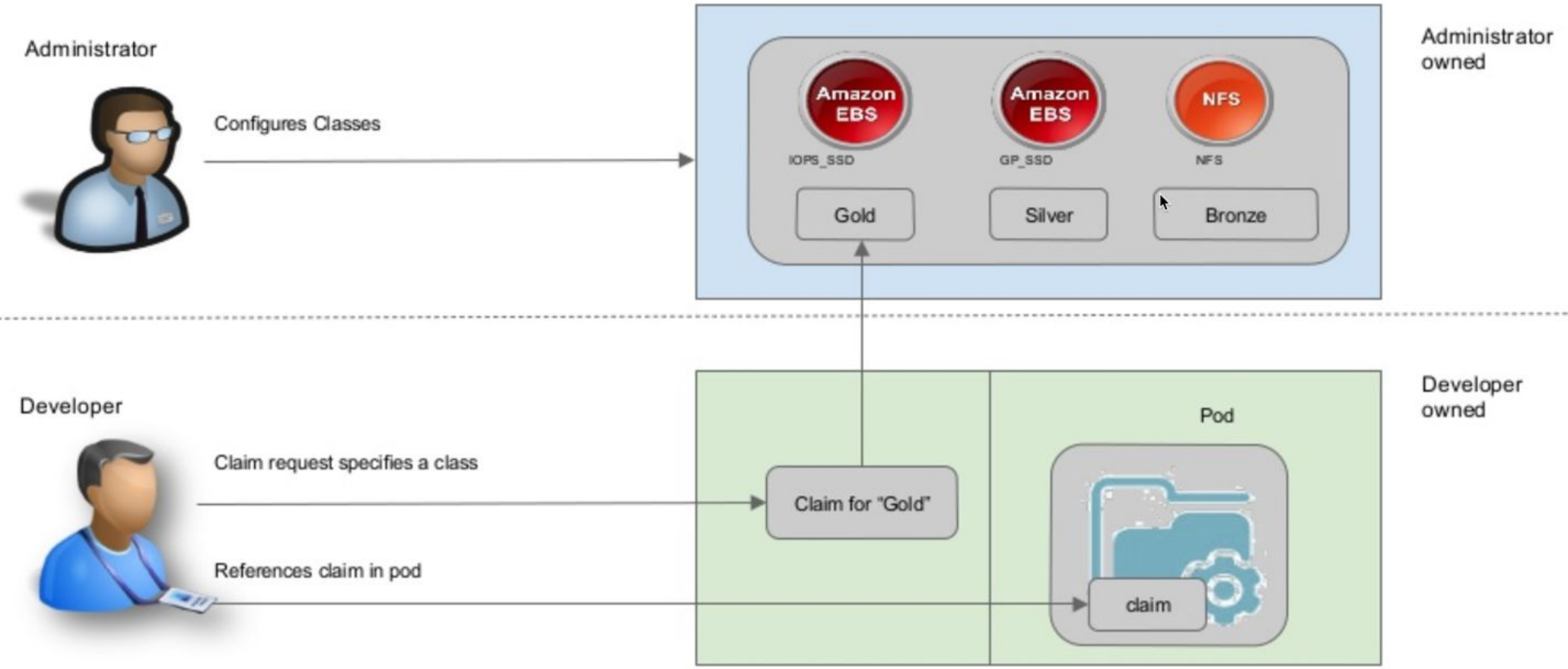
Persistent Volume

- 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

- 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
```

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

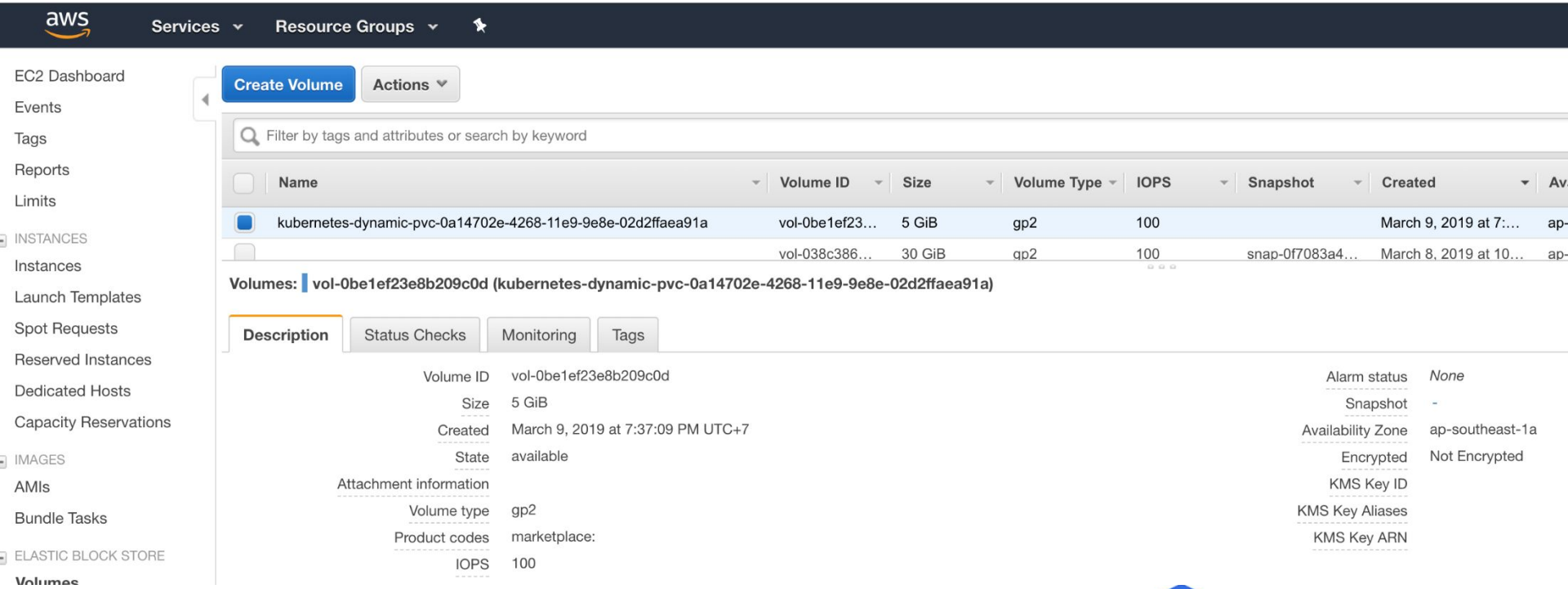
- Dynamic Provision
  - Test create SC/PVC

```
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

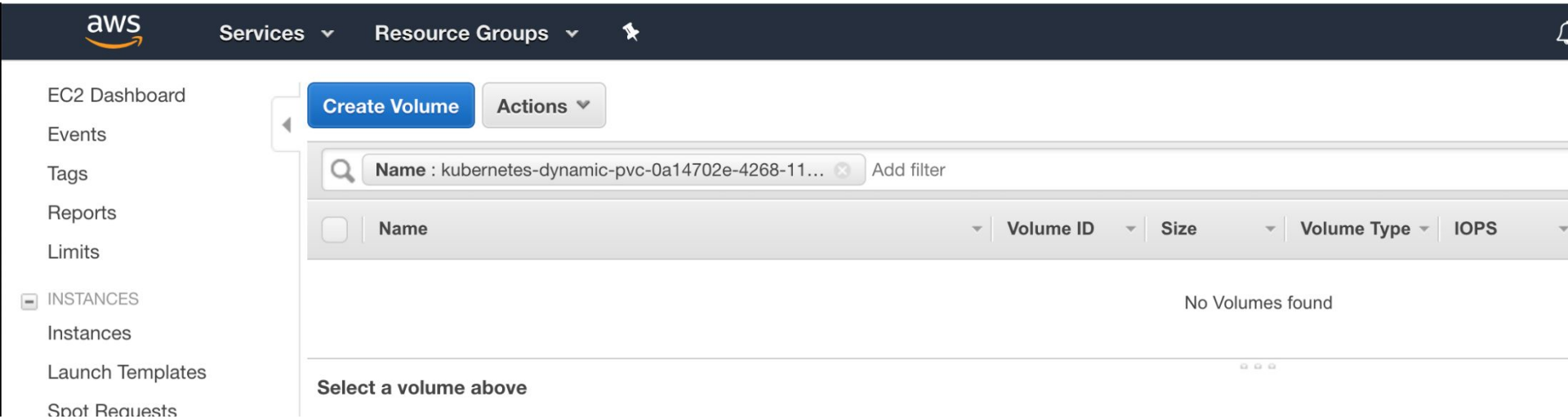
Persistent Volume Claims

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”

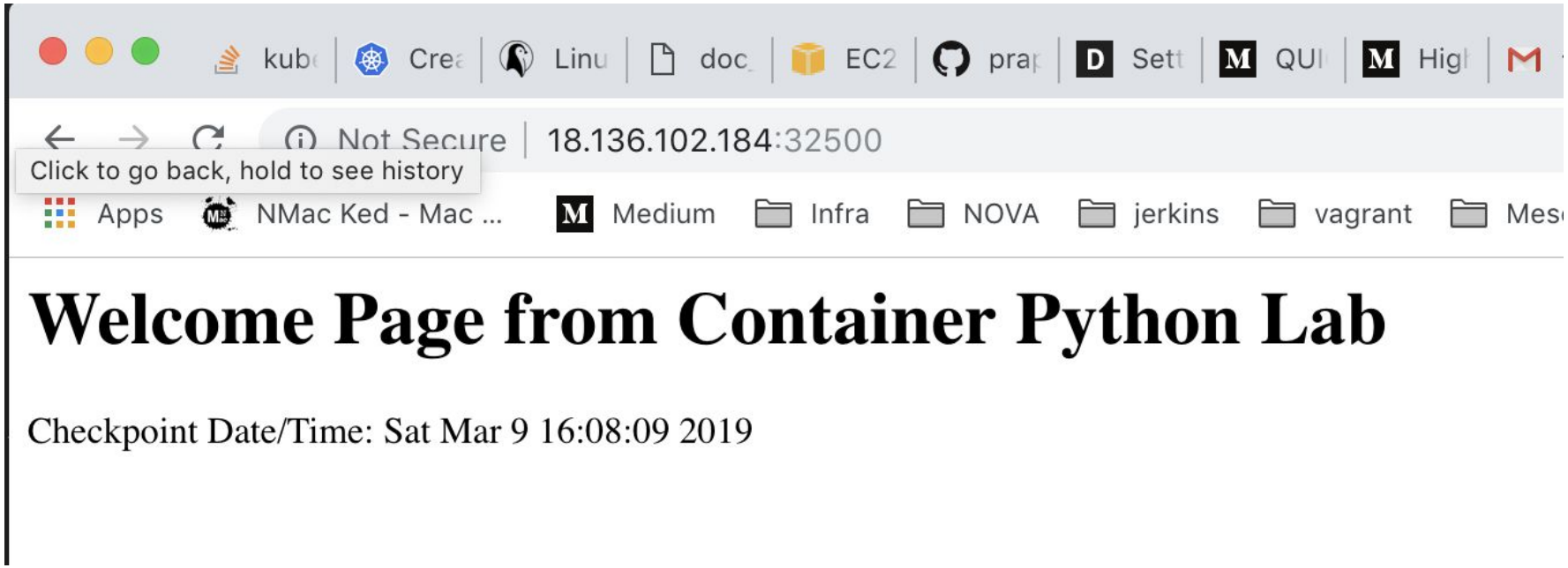
Persistent Volume

- 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

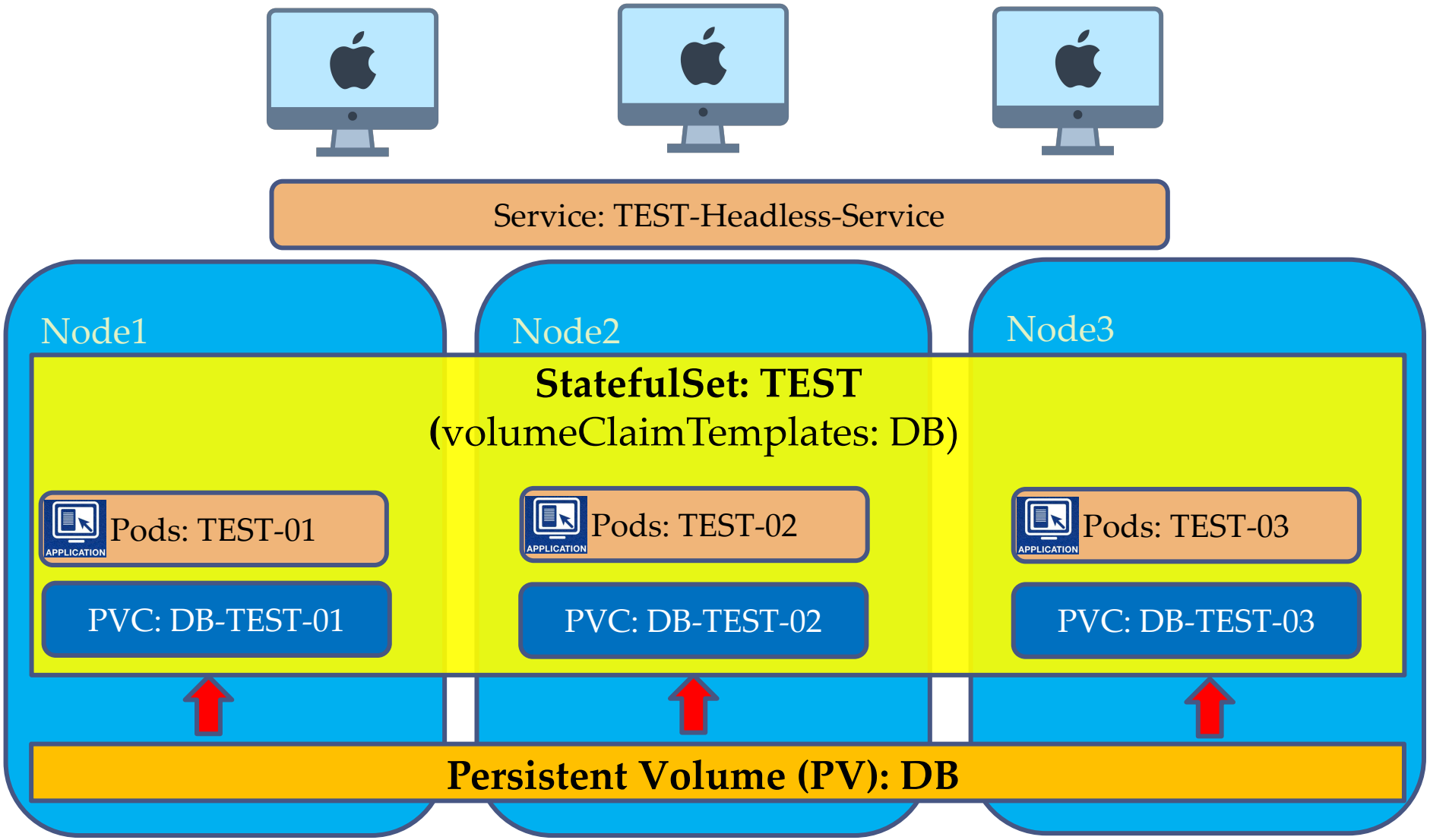
StatefulSet

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
```

Recap Day 5

- 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

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