



微服務基礎概念

Alfie Chen 陳逸凡

2018/04/27

✉️ alfie.c@inwinstack.com

🐦 alfieYFC 📱 alfie.yfc

🐦 inwinSTACK 📱 inwinSTACK

📺 inwinSTACK 📱 Groups/k8s.tw

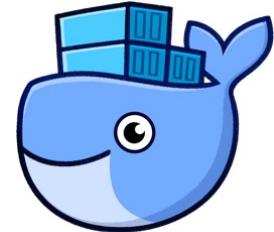


kubernetes

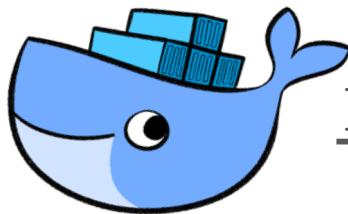
Agenda



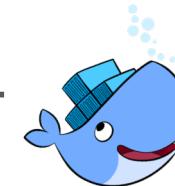
World of Microservices



Lab#1 Docker



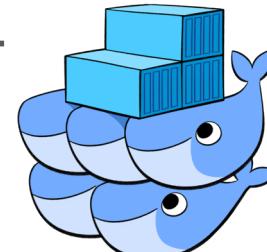
Intro to Kubernetes



Lab#2 Minikube



Demo & Case Studies



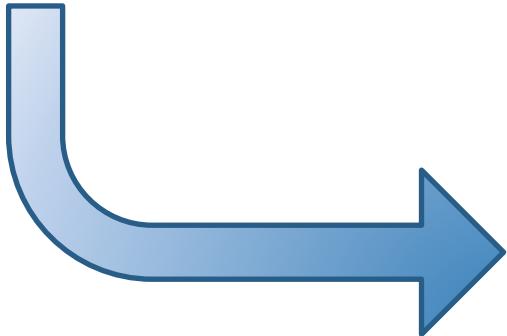
World of Microservices



Virtual Machines, Containers, etc.

App Development Changes

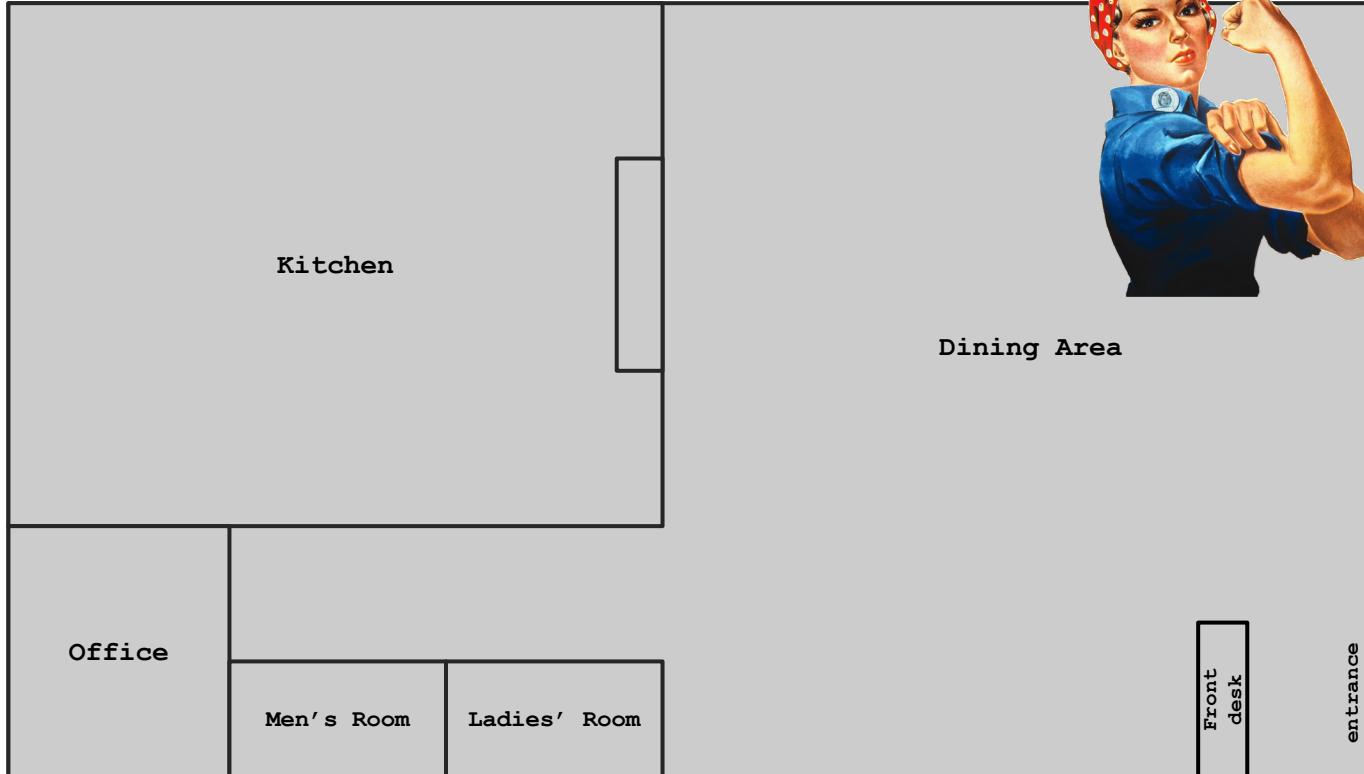
- Huge Applications
- Hours to Build
- Huge Downtime



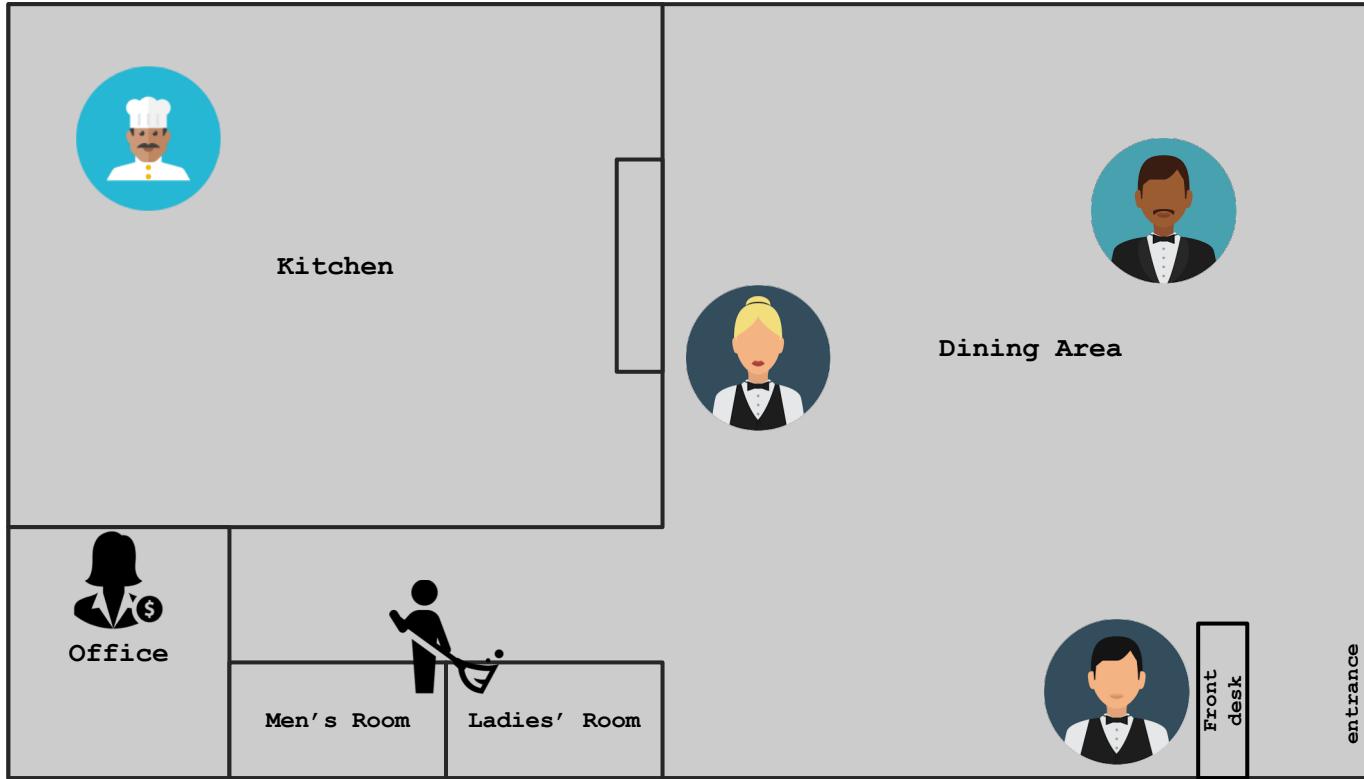
- Microservices
- Container Deployment
- DevOps
- Continuous Integration
- Continuous Delivery



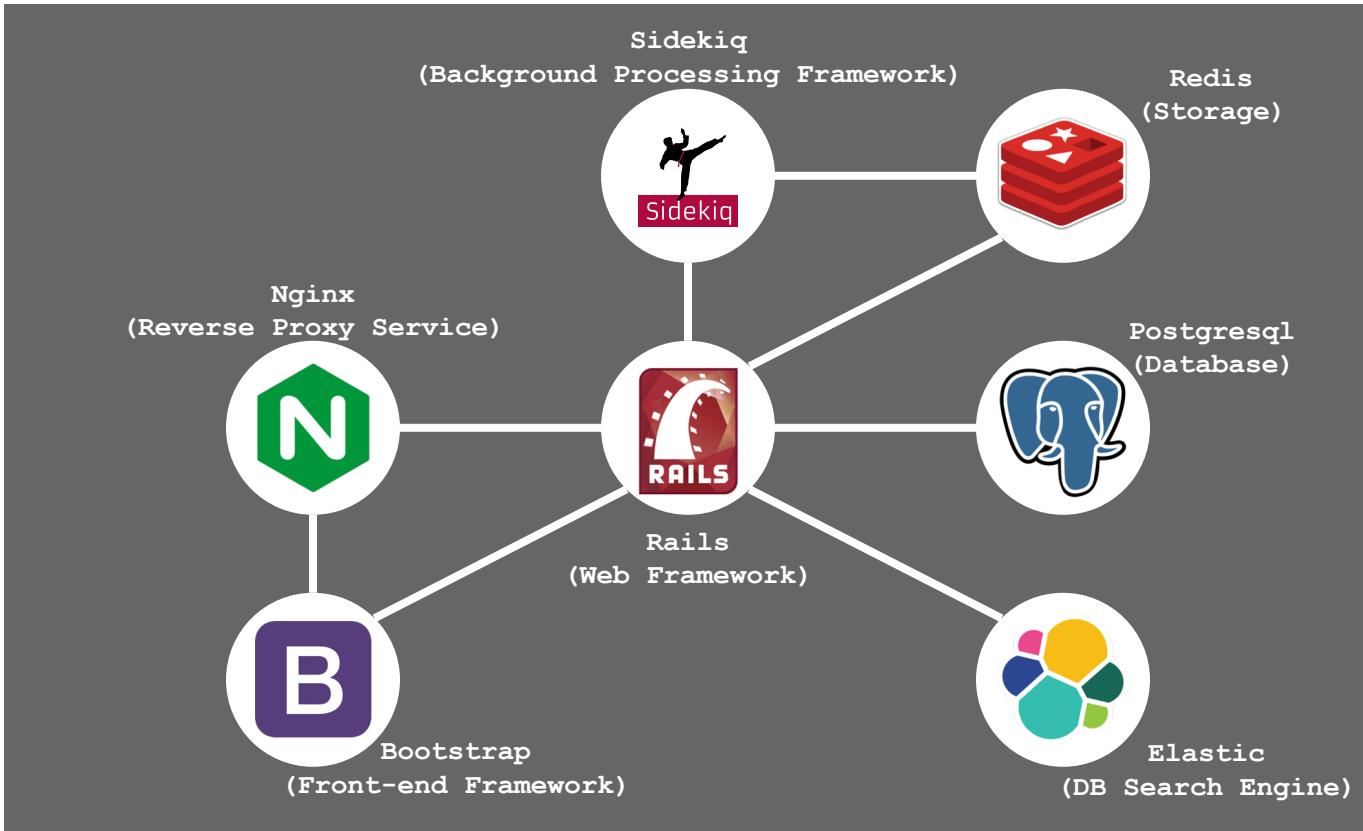
What is Microservices?



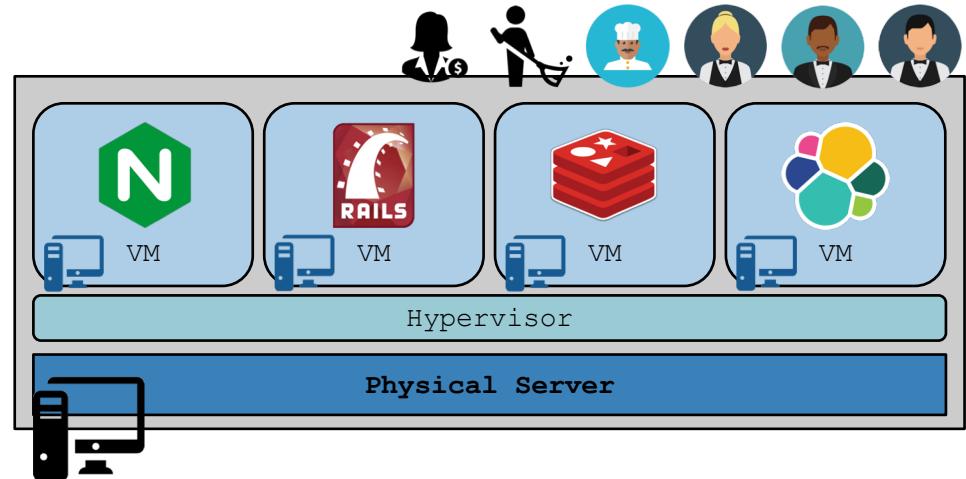
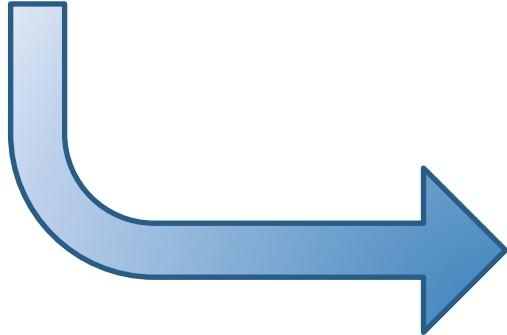
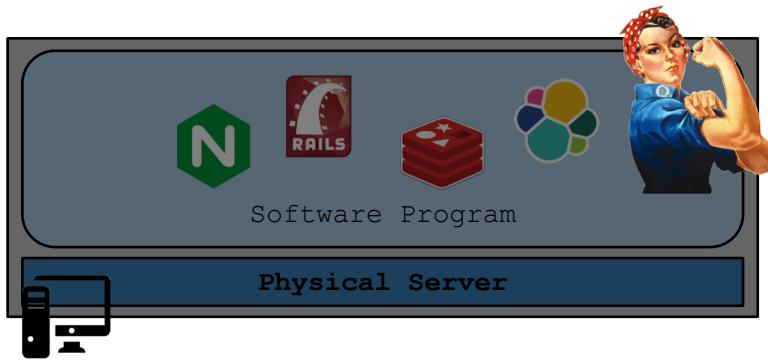
What is Microservices?



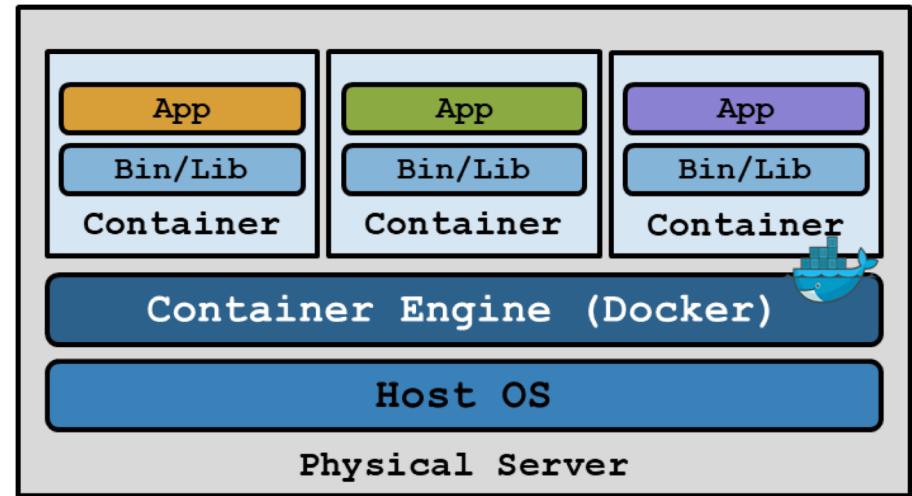
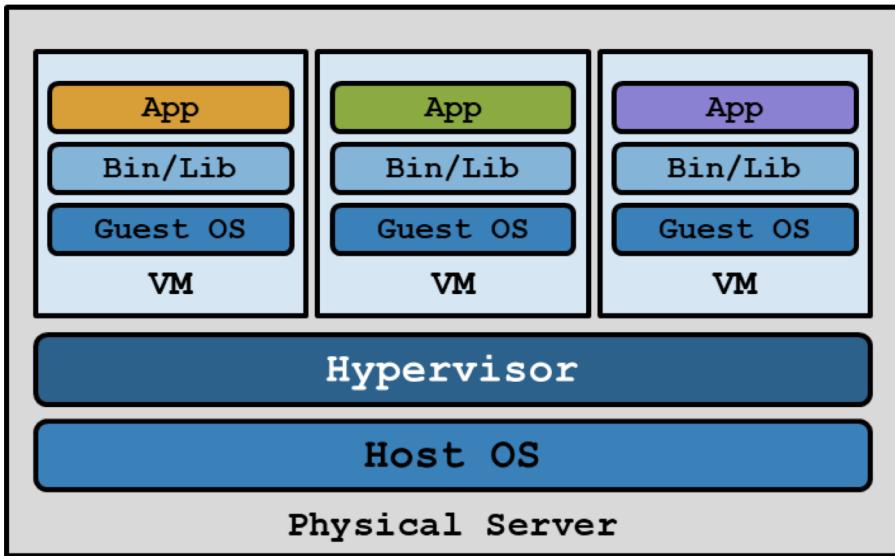
What is Microservices?



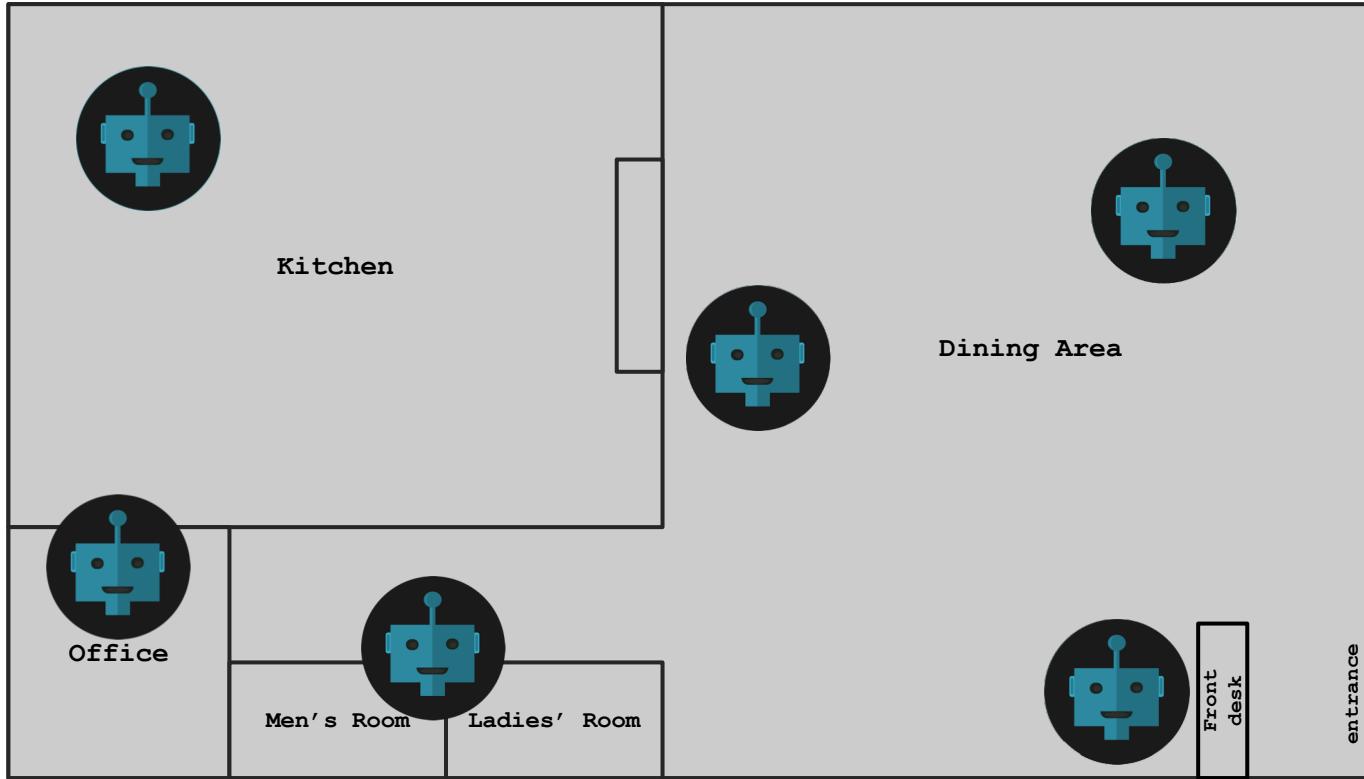
What is Microservices?



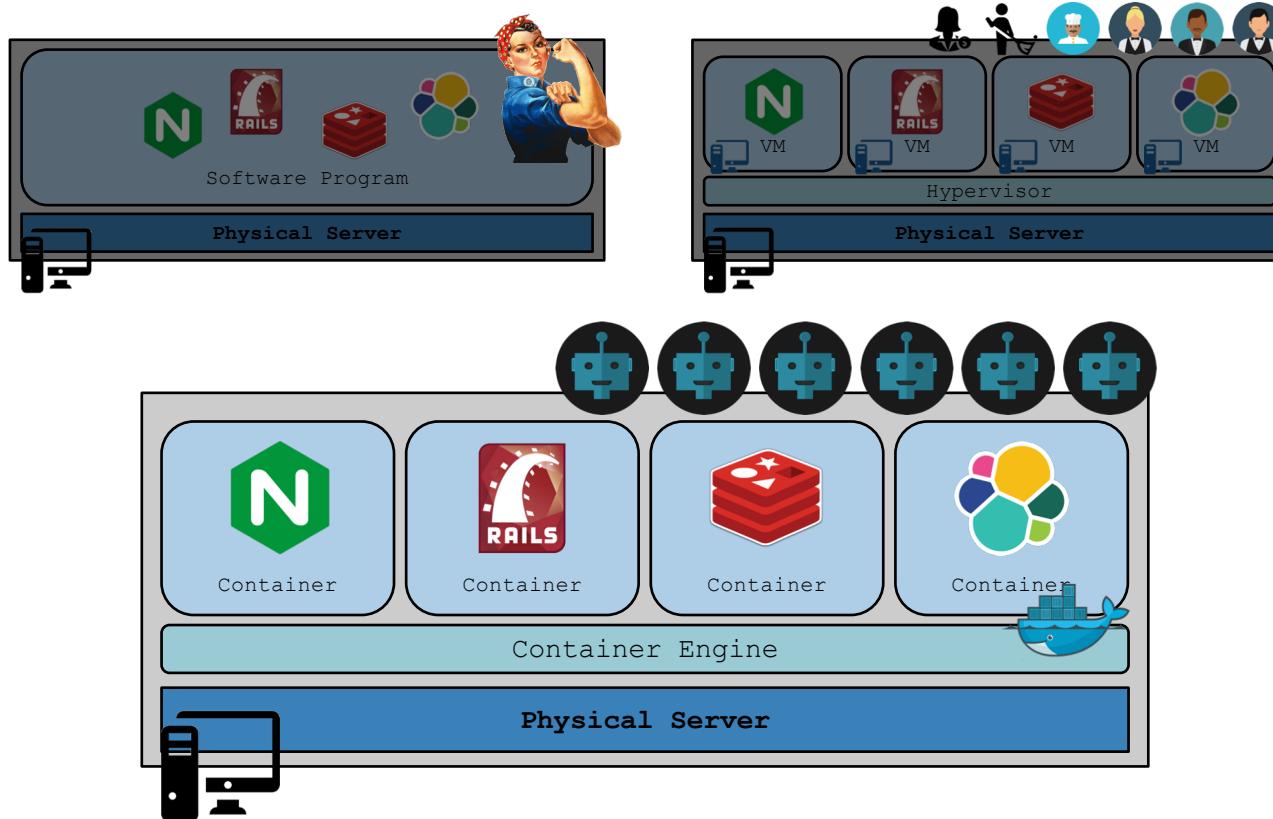
What is Microservices?



What is Microservices?

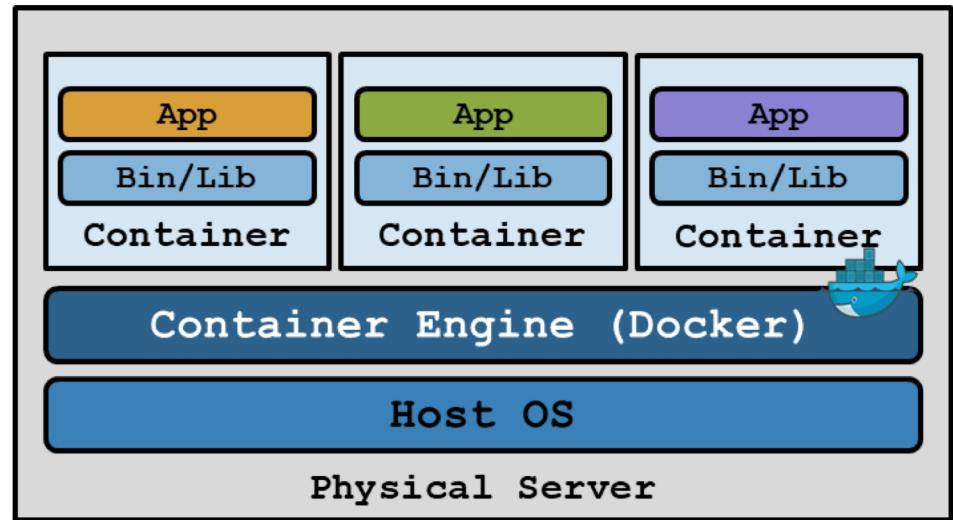


What is Microservices?

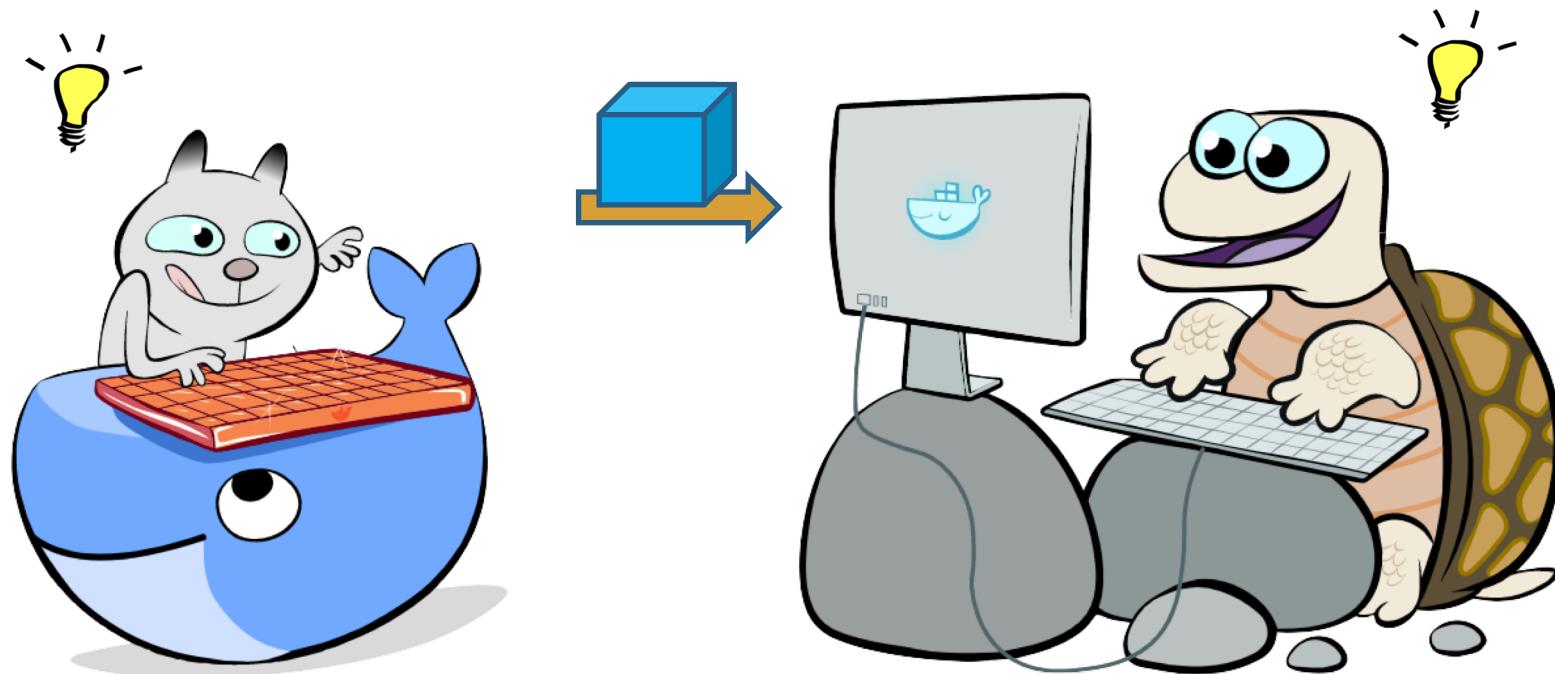


Containers

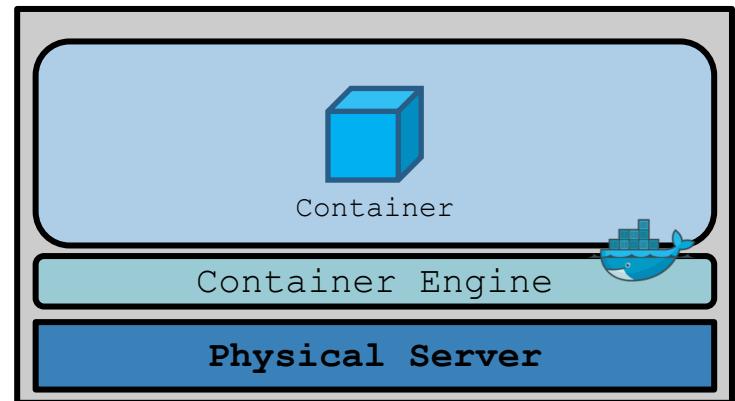
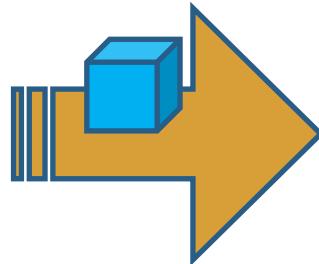
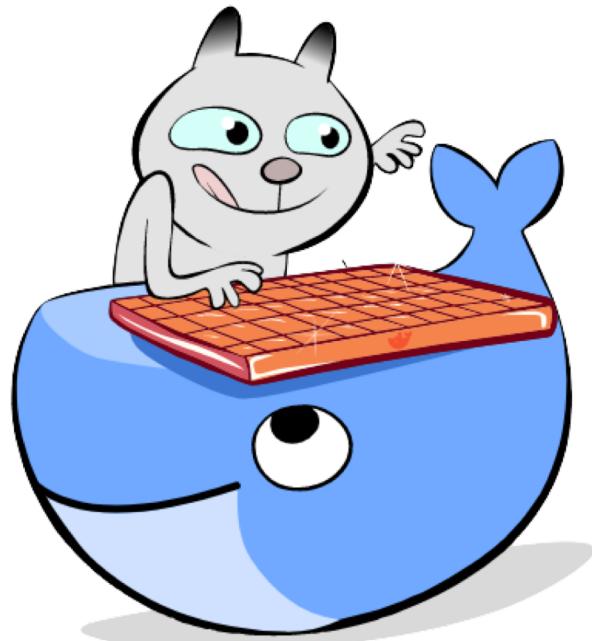
- Light-weighted
- Isolated Apps
- Simple Version Control
- Fast Delivery
- Portability



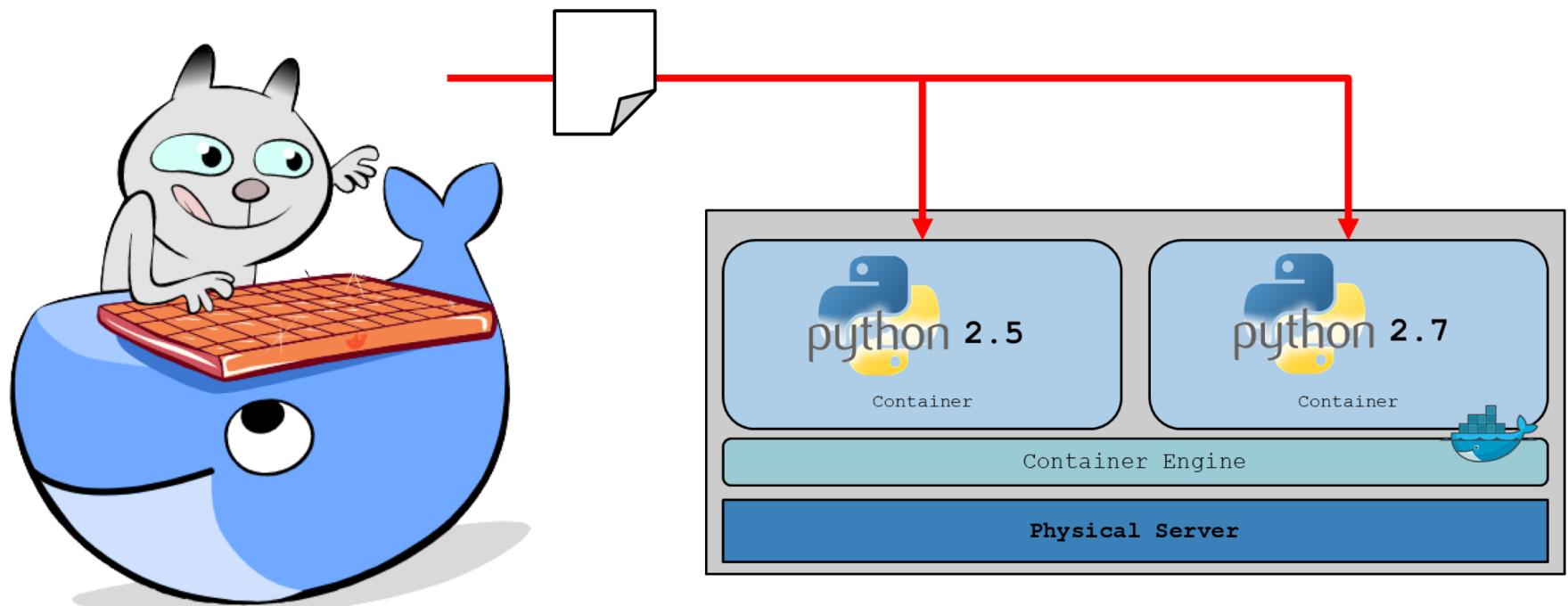
Containers



Containers – Local Development



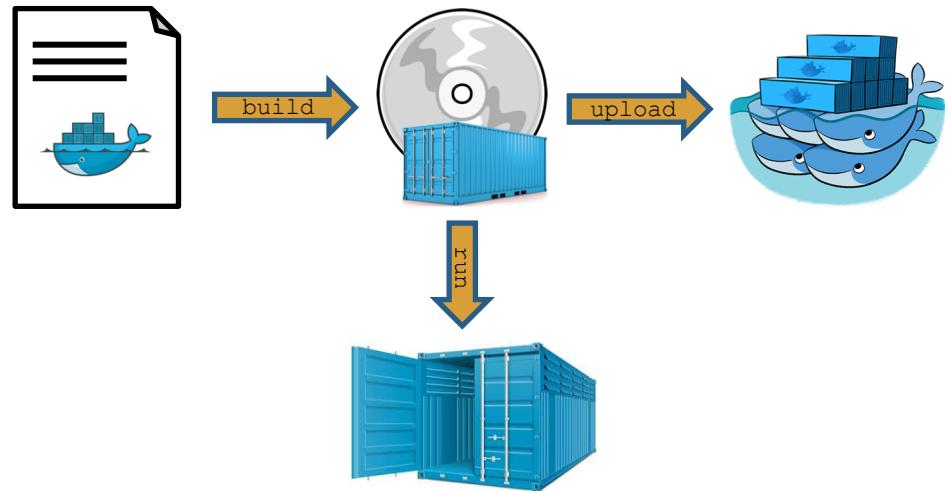
Containers – Simplified Testing



Docker

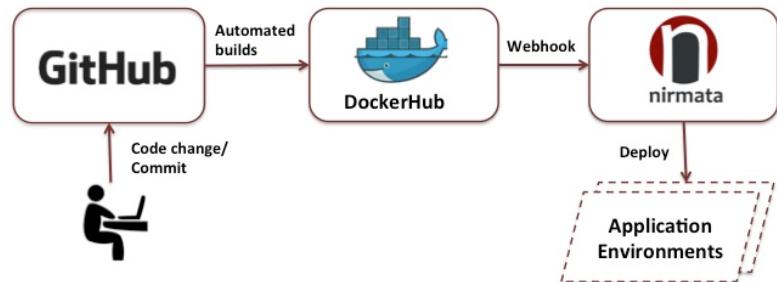
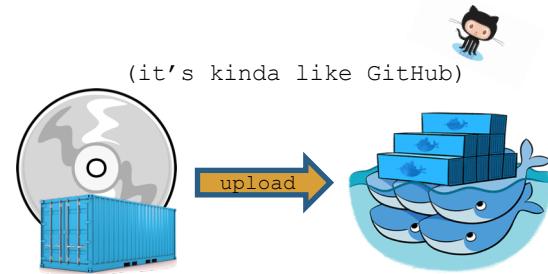
Docker provides a uniformed wrapper around a software package

- Containers
- Images
- Dockerfile
- Docker Registry



Docker Hub / Store

- <https://hub.docker.com/>
- <https://store.docker.com/>
- **Public** Docker Registry
- Additional Features
 - 1. Automated Builds
 - 2. Webhooks
 - 3. Organizations



<https://github.com/acinwinstack/minikube-intro-workshop>

Lab#1 Docker



Installation; HelloWorld; Docker Commands!

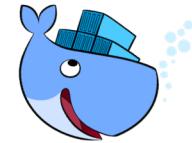
Lab1.1 - Installation (Windows)

- <https://docs.docker.com/engine/installation/>
- <https://download.docker.com/win/stable/Docker%20for%20Windows%20Installer.exe>
- Windows 10 Pro with Hyper-V
- Windows Server 2016



Lab1.1 - Installation (Mac)

- <https://docs.docker.com/engine/installation/>
- <https://docs.docker.com/docker-for-mac/install/>
- Mac 2010 hardware model or newer
- MacOS El Capitan 10.11 or newer
- At least 4GB RAM

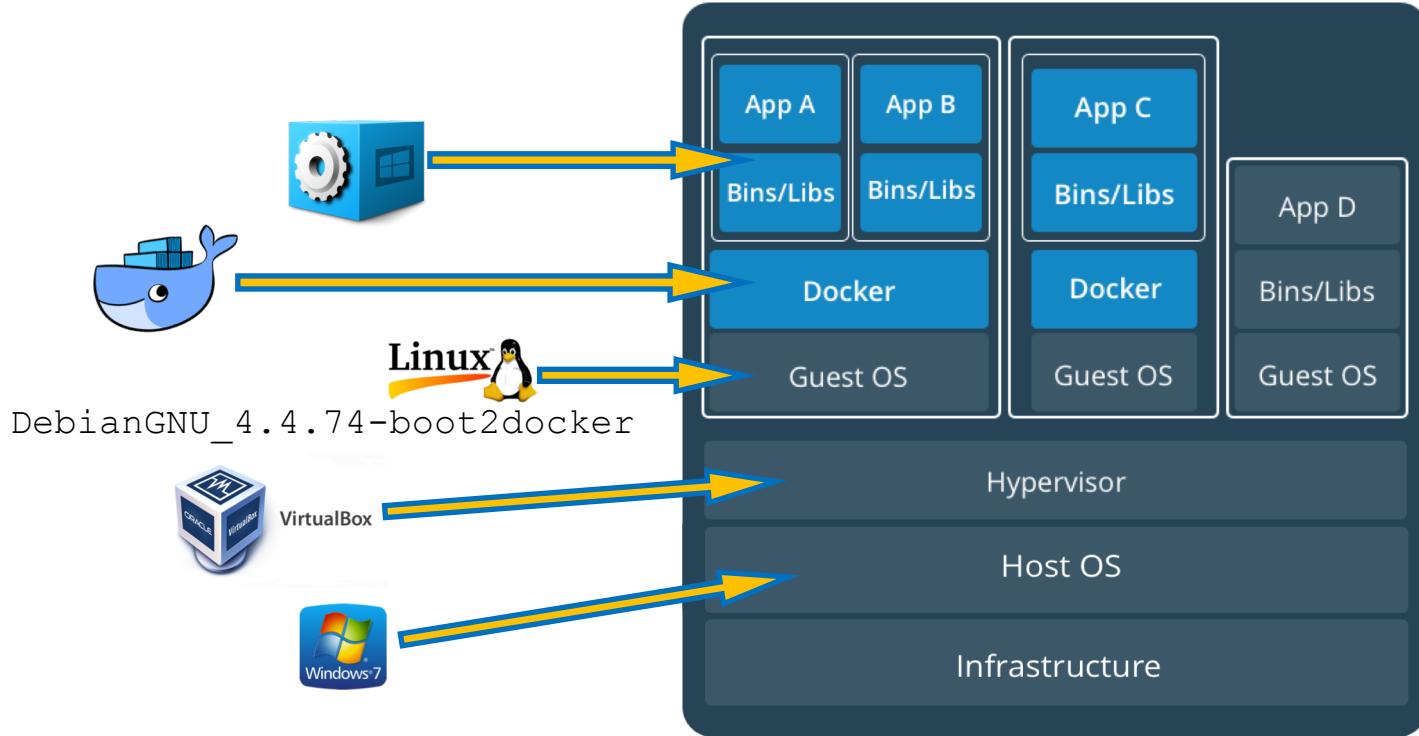


Lab1.1 - Installation (Toolbox)

- <https://docs.docker.com/engine/installation/>
- <https://docs.docker.com/toolbox/overview/>
- Any Windows system Docker for Windows doesn't support
- Any Mac system Docker for Mac doesn't support
- Docker Toolbox {
 git
 virtualbox



Lab1.1 - Installation (Toolbox)



Lab1.1 - Installation (Linux)

- <https://docs.docker.com/install/#server>
- Enterprise Edition     
- Community Edition    

The easy way (CE) :

```
curl -fsSL "https://get.docker.com/" | sh  
sudo systemctl start docker
```



Lab1.2 – Docker Commands

```
$ docker pull nginx
$ docker login
$ docker tag nginx <username>/nginx
$ docker push <username>/nginx
$ docker run -itd nginx
$ docker ps
$ docker run -itd -p 80:80 --name test nginx:1.12.2
$ curl localhost
$ docker rm -f <containerID>
```



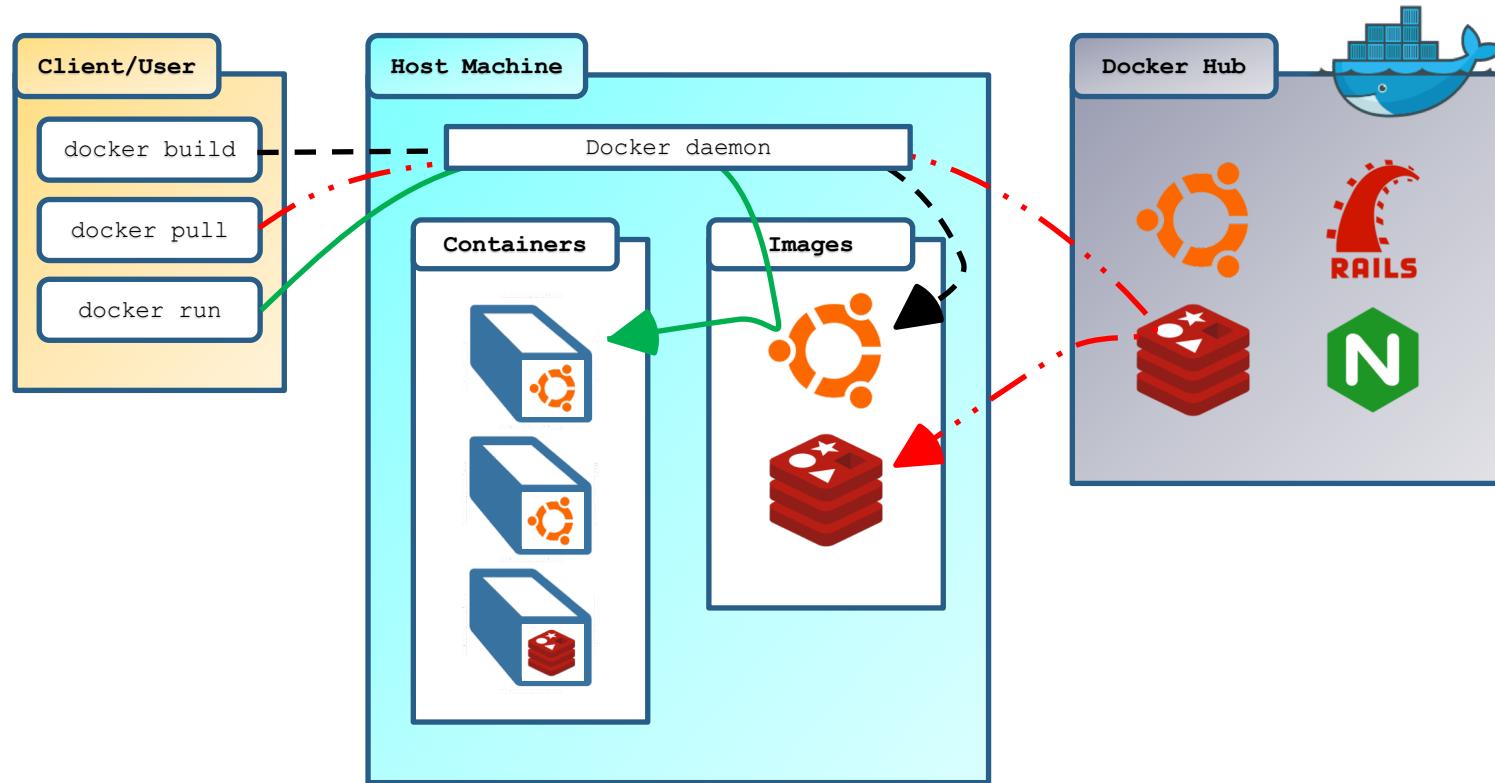
Lab1.2 – Docker Commands

```
$ git clone https://github.com/acinwinstack/minikube-intro-workshop.git  
$ cd minikube-intro-workshop/
```

```
$ docker build -t <username>/helloworld:v1 ./build/  
$ docker images  
$ docker run -itd --name test2 -p 8080:80 \  
  <username>/helloworld:v1  
$ curl localhost:8080
```

```
$ docker push <username>/helloworld:v1
```

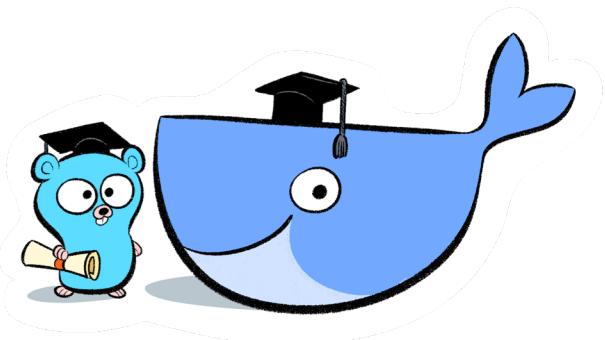
How does it work?



Commands

Commands:

attach	Attach local standard input, output, and error streams to a running container
build	Build an image from a Dockerfile
commit	Create a new image from a container's changes
cp	Copy files/folders between a container and the local filesystem
create	Create a new container
diff	Inspect changes to files or directories on a container's filesystem
events	Get real time events from the server
exec	Run a command in a running container
export	Export a container's filesystem as a tar archive
history	Show the history of an image
images	List images
import	Import the contents from a tarball to create a filesystem image
info	Display system-wide information
inspect	Return low-level information on Docker objects
kill	Kill one or more running containers
load	Load an image from a tar archive or STDIN
login	Log in to a Docker registry
logout	Log out from a Docker registry
logs	Fetch the logs of a container
pause	Pause all processes within one or more containers
port	List port mappings or a specific mapping for the container
ps	List containers
pull	Pull an image or a repository from a registry
push	Push an image or a repository to a registry
rename	Rename a container
restart	Restart one or more containers
rm	Remove one or more containers
rmi	Remove one or more images
run	Run a command in a new container
save	Save one or more images to a tar archive (streamed to STDOUT by default)
search	Search the Docker Hub for images
start	Start one or more stopped containers
stats	Display a live stream of container(s) resource usage statistics
stop	Stop one or more running containers
tag	Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
top	Display the running processes of a container
unpause	Unpause all processes within one or more containers
update	Update configuration of one or more containers
version	Show the Docker version information
wait	Block until one or more containers stop, then print their exit codes



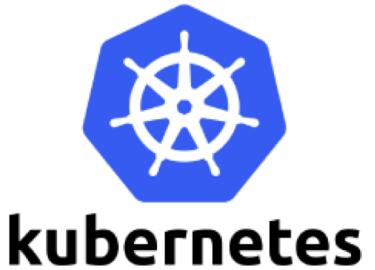
Intro to Kubernetes



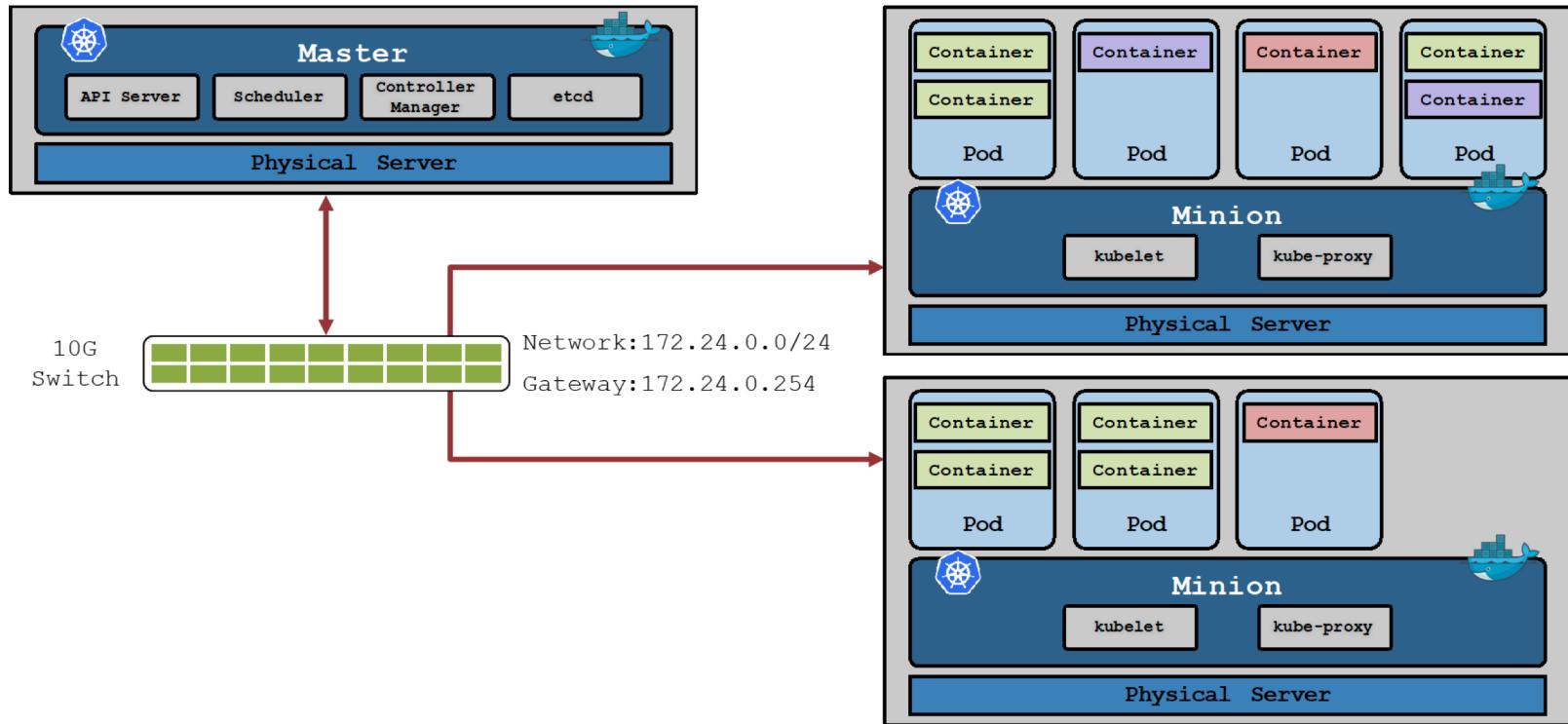
A powerful container orchestrator open sourced by Google Inc.

Introduction

- Κυβερνήτης: Governor, Pilot.
- “K8s”
- Current Release: 1.10+
- <https://kubernetes.io/>
- Release Roadmap
<https://github.com/kubernetes/kubernetes/milestones/>

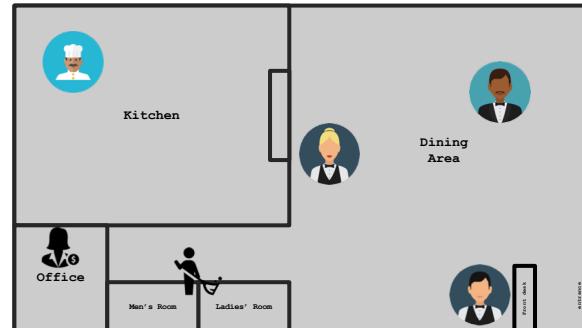
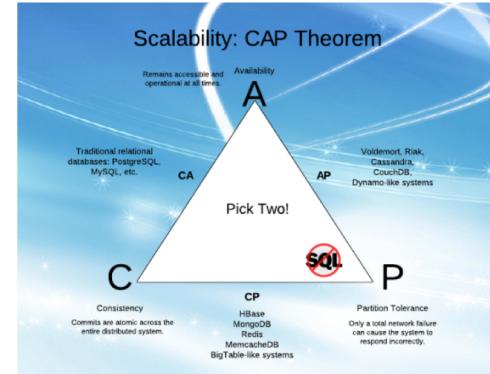


Kubernetes Cluster



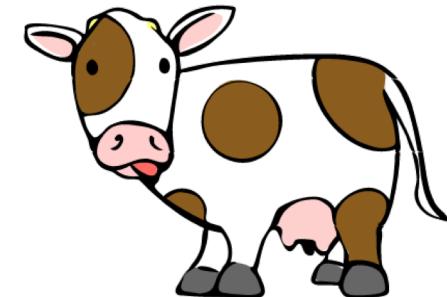
Kubernetes Concepts

- Controllers
 - Deployment
 - Stateful Set
 - Daemon Set
 - Job
- Service



Kubernetes Concepts - Controllers

- Deployment
 - Stateless
 - Randomly assigned identities
 - Availability > Consistency
 - **[Web Frontends]**



Kubernetes Concepts - Controllers

- Stateful Set
 - Keeps states
 - Persistent data
 - Each has its own identity
 - Availability < Consistency
 - **[Databases]**
 - **[Message Queues]**



Kubernetes Concepts - Controllers

- Daemon Set
 - Background process
 - One pod per node
 - Node labels
 - **[Cluster Storage]**
 - **[Logging]**
 - **[Node Monitoring]**

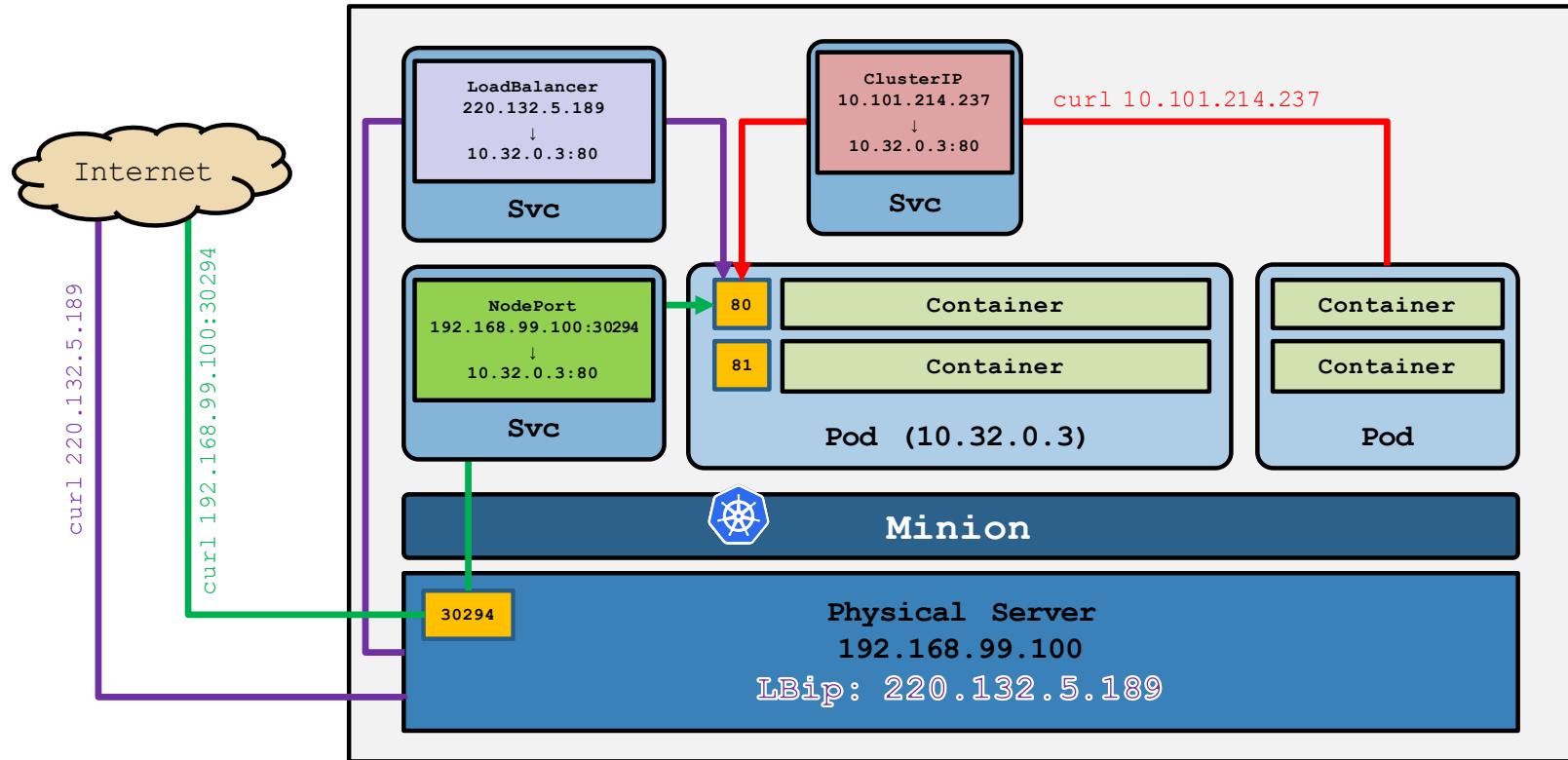


Kubernetes Concepts - Controllers

- [Jobs](#)
 - Batch Pattern
 - Run in parallel
 - Run to completion (then exit)
 - Independent but related processes



Kubernetes Concepts – Service



Concepts

- ▶ Overview
- ▶ Kubernetes Architecture
- ▶ Extending the Kubernetes API
- ▶ Containers
- ▼ Workloads
 - ▶ Pods
 - ▼ Controllers
 - Replica Sets
 - Replication Controller
 - Deployments
 - StatefulSets
 - PetSets
 - Daemon Sets
 - Garbage Collection
 - Jobs - Run to Completion
 - Cron Jobs
- ▶ Configuration
- ▶ Services, Load Balancing, and Networking
- ▶ Storage
- ▶ Cluster Administration

Kubernetes Concepts

There's more!!

So many tools to choose from....

<https://kubernetes.io/docs/concepts/>

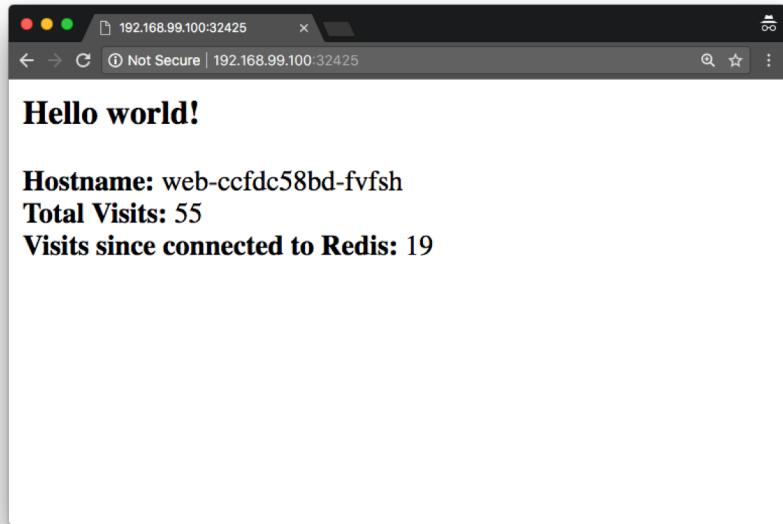
Lab#2 Minikube



Local, single node K8s cluster

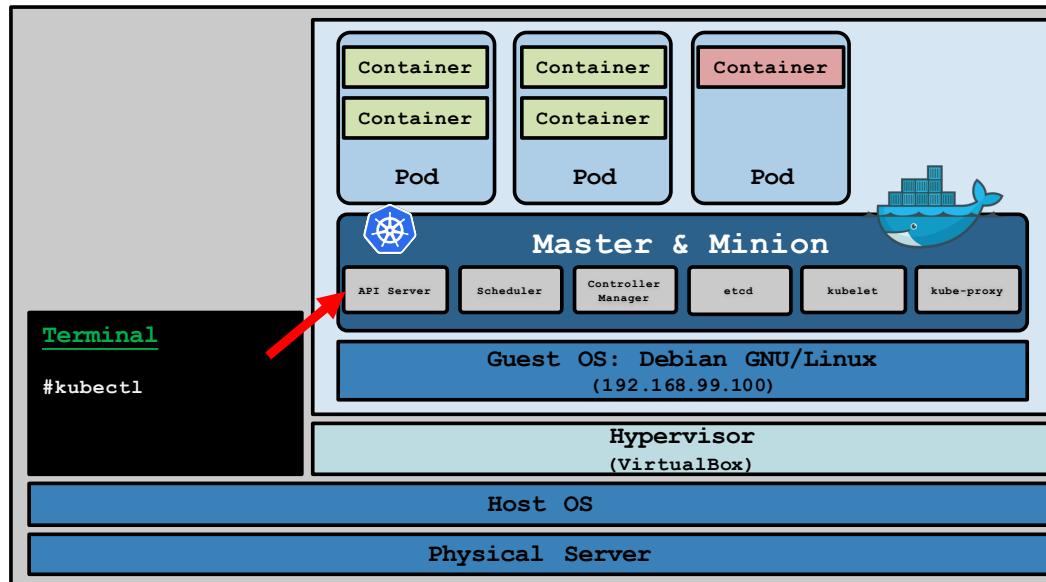
Lab Description

- Single Node
- Persistent Storage
- Rolling update



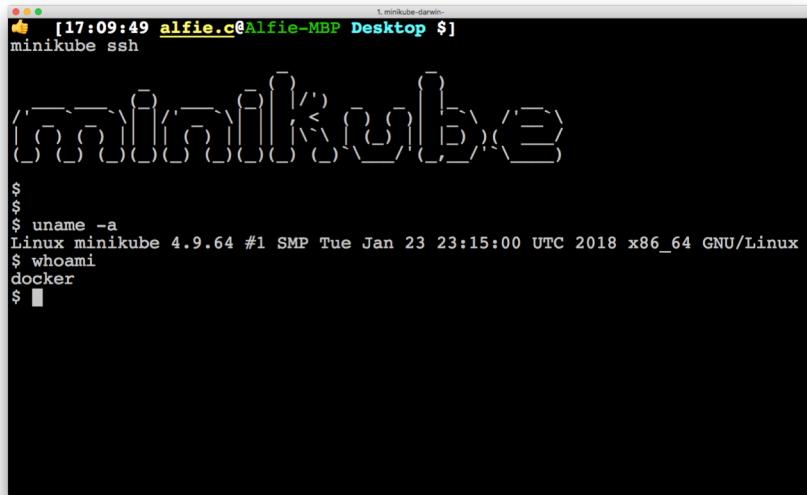
Lab2.1 - Installation

<https://kubernetes.io/docs/tasks/tools/install-minikube/>



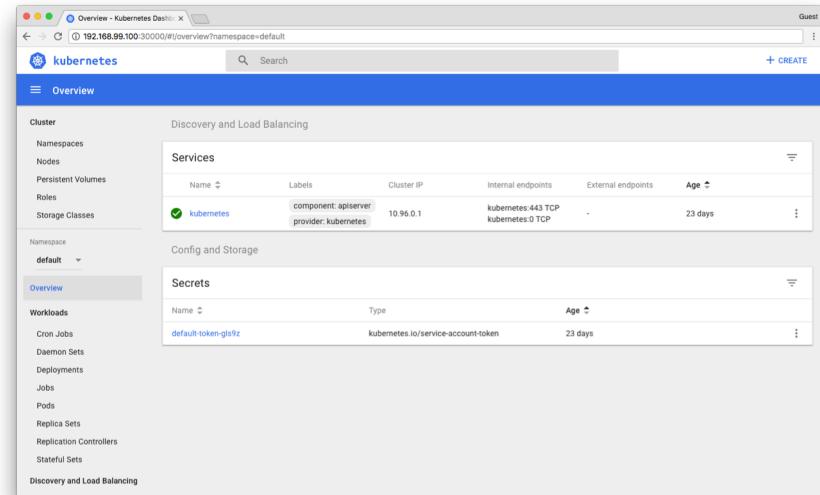
Lab2.2 – Minikube Commands

```
$ minikube start  
$ minikube dashboard  
$ minikube ssh  
$ minikube stop
```



A terminal window titled "minikube-darwin" showing the following command history:

```
[17:09:49 alfie.c@Alfie-MBP Desktop $]  
$ minikube ssh  
$  
$ uname -a  
Linux minikube 4.9.64 #1 SMP Tue Jan 23 23:15:00 UTC 2018 x86_64 GNU/Linux  
$ whoami  
docker  
$
```



The Kubernetes Dashboard Overview page at 192.168.99.100:30000/#/overview?namespace=default shows the following data:

Name	Labels	Cluster IP	Internal endpoints	External endpoints	Age
kubernetes	component: apiserver provider: kubernetes	10.96.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	23 days

Secrets table:

Name	Type	Age
default-token-gls9z	kubernetes.io/service-account-token	23 days

Lab2.3 - Run the Web App

```
$ kubectl apply -f lab1/redis_pod.yml  
$ kubectl get po -o wide  
write down the pod IP of redis
```

```
$ vi lab1/web_deploy.yml  
Config REDIS_IP
```

```
$ kubectl apply -f lab1/web_deploy.yml
```

Lab2.4 – Expose the Web App

```
$ kubectl expose deploy hello-world-deploy \
  --name hello-world-svc \
  --type NodePort \
  --port 80 \
  --target-port 80
$ kubectl get svc | write down the NodePort IP of hello-world-svc
```

```
$ curl <minikube>:NodePortIP
```

Lab2.5 – Persistent Volume

```
$ vi lab2/web_deploy.yml
```

```
Config REDIS_IP
```

```
$ kubectl apply -f lab2/web_deploy.yml
```

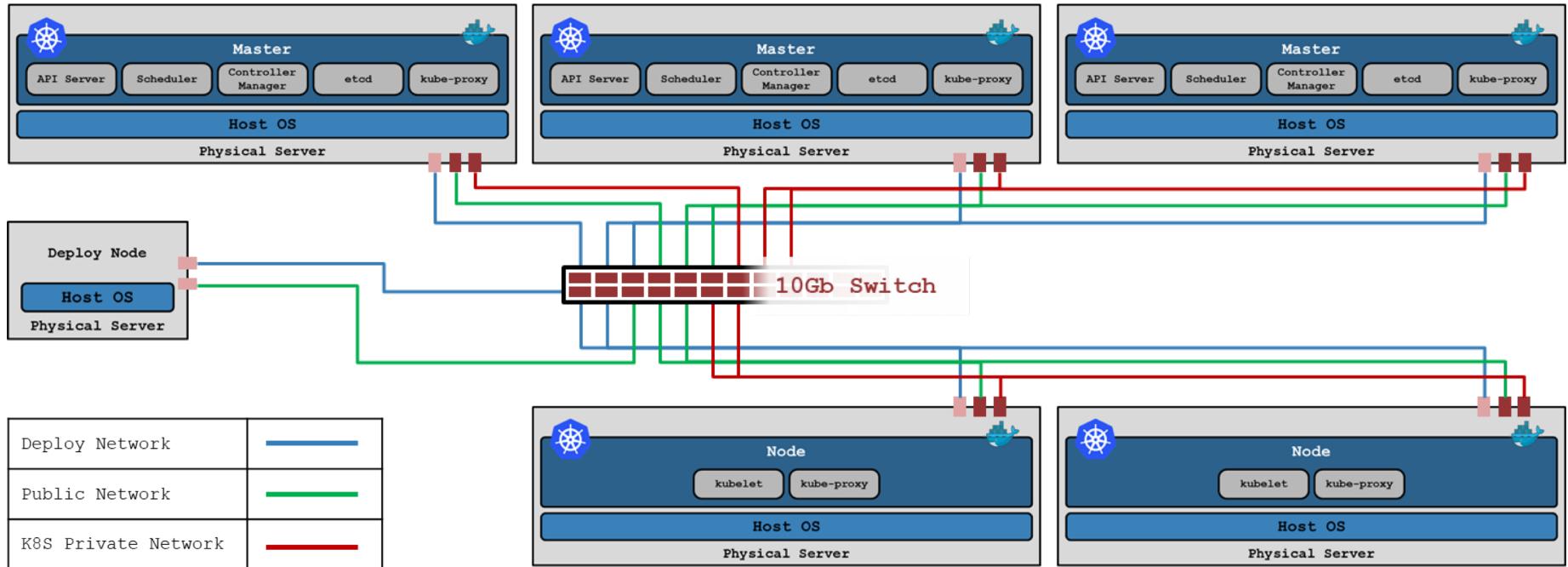
Production Solutions & Case Studies

Infrastructure designs for production, with a
successful story to tell

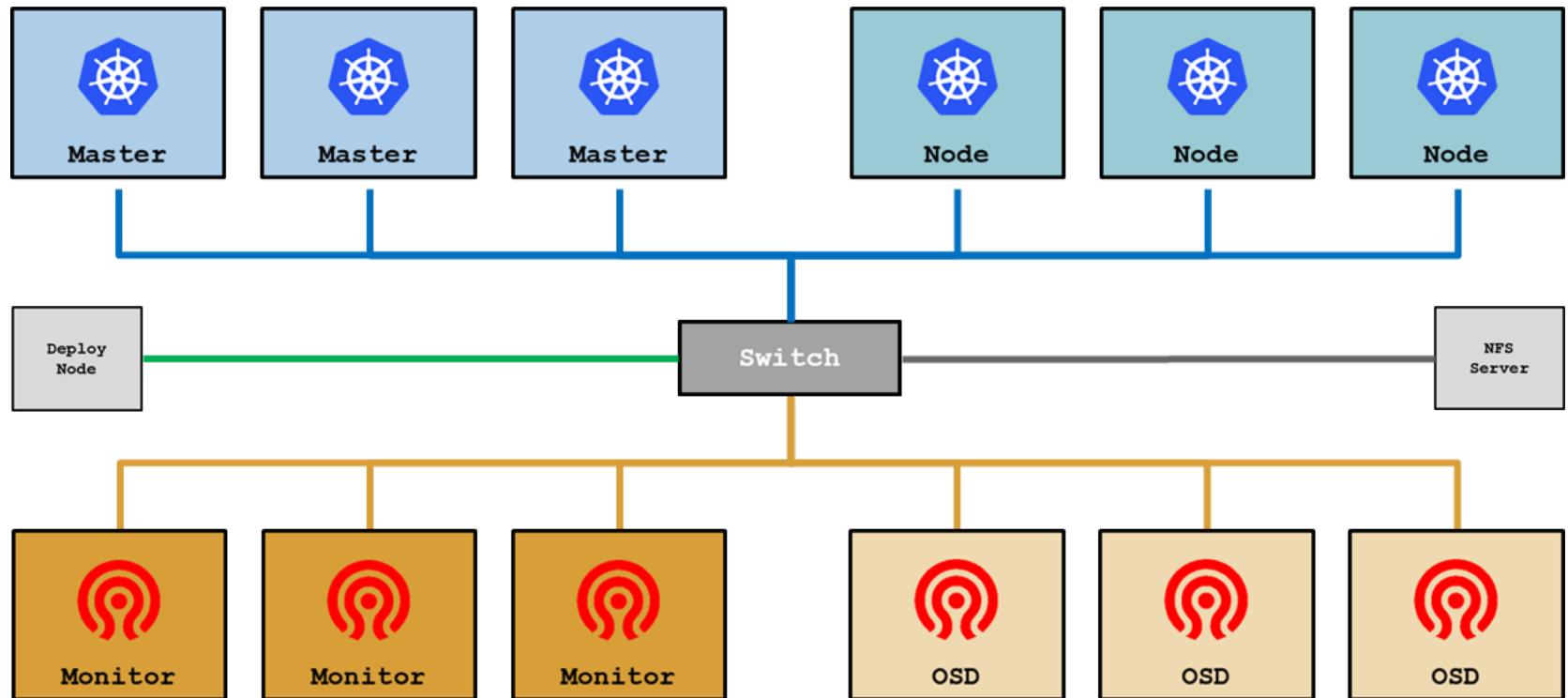
K8s Minimum Specs

節點角色(主機名稱)	硬體項目	規格
Kubernetes Master	CPU	8 Core
	RAM	32GB
	HDD	1TB SATA
	NIC	10Gb ports
Kubernetes Node	CPU	16 Core
	RAM	64GB
	HDD	1TB SATA
	NIC	10Gb ports

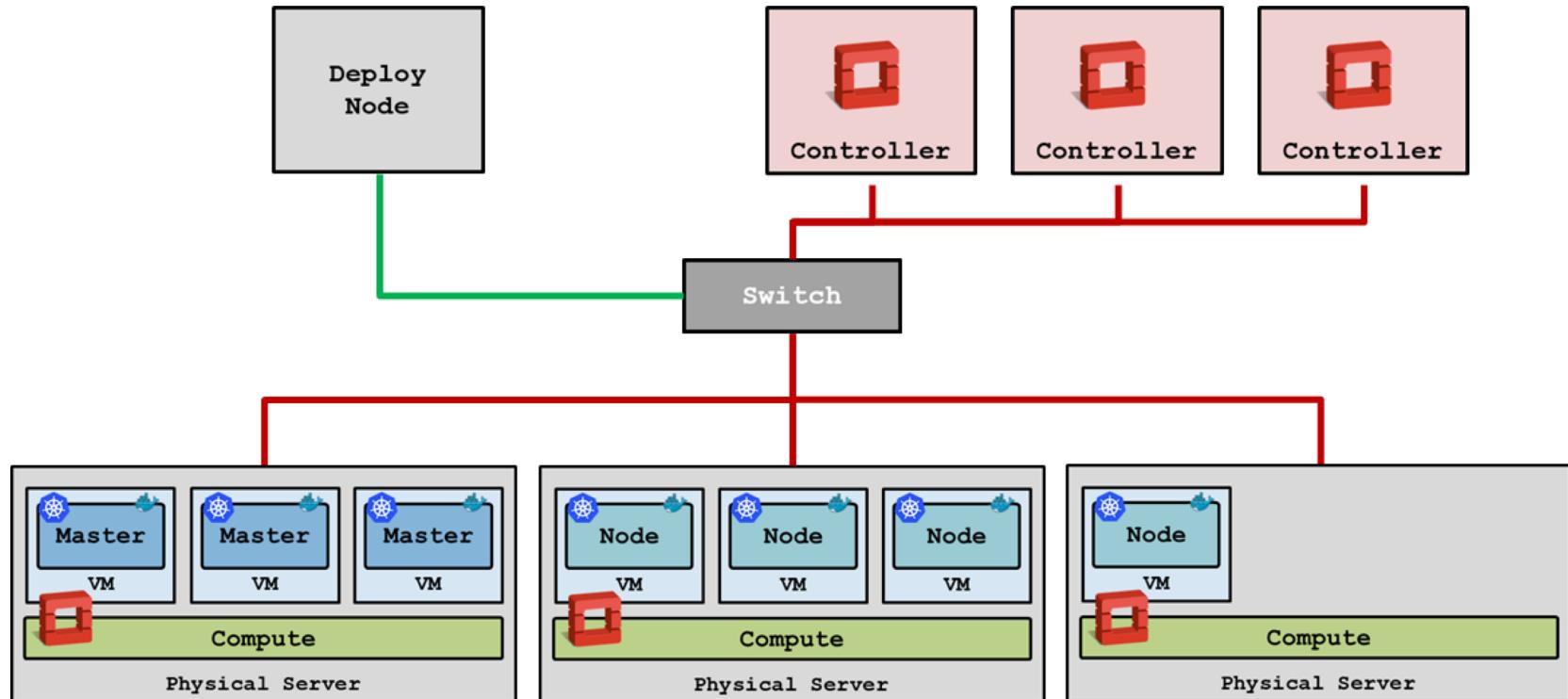
K8s Minimum Architecture (HA)



Kubernetes + Ceph

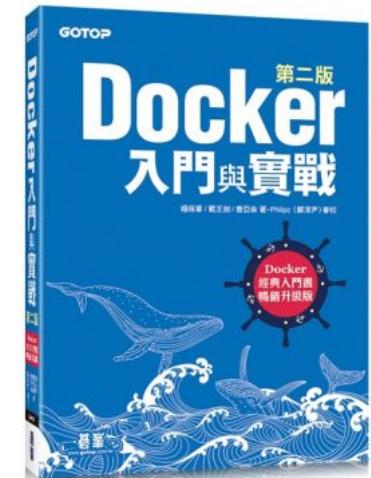
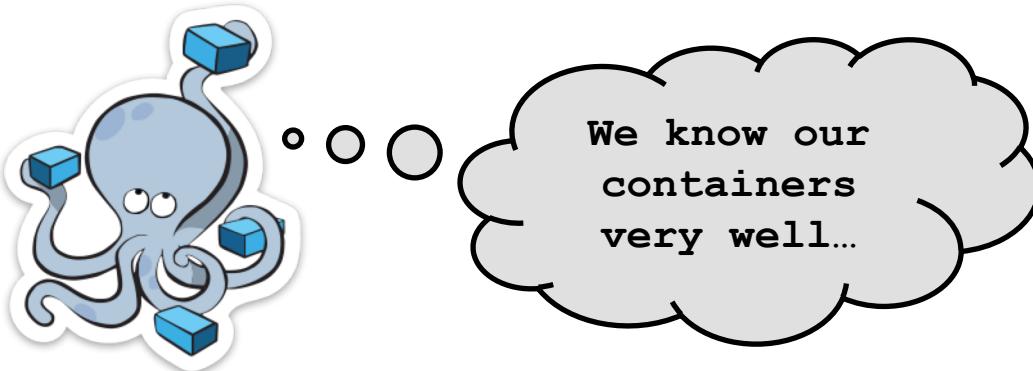


Virtual Kubernetes



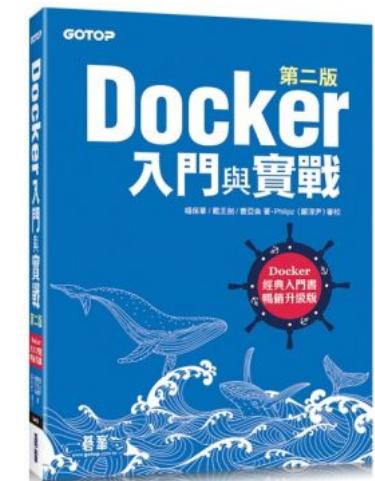
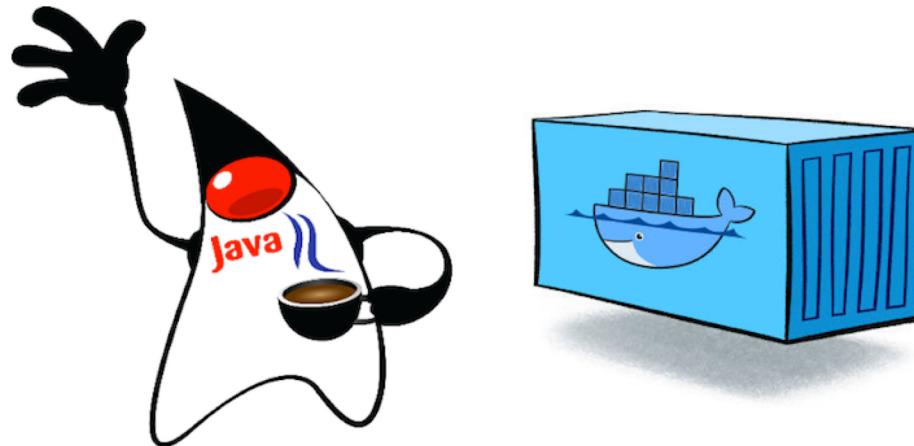
A Successful Story – User Background

- A Hospital in Taipei
- Java developers
- Already using Docker containers
- CI/CD practice with GitLab + Docker



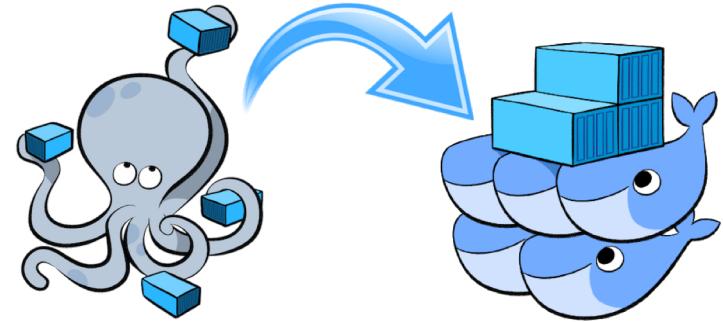
A Successful Story - User Background

- EMR System Applications
- HR Software Applications
- Radiology Information System Applications



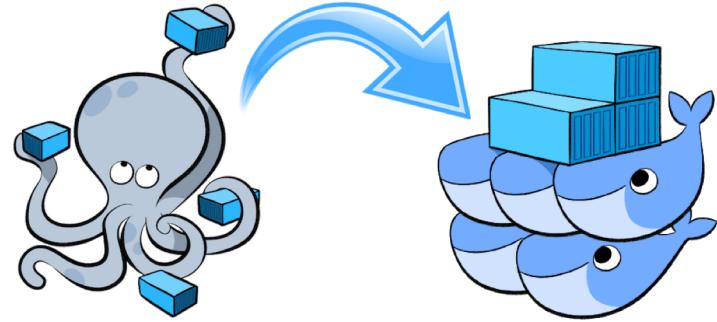
The Challenge

- Docker Swarm is not enough
- Require auto-scaling
- Require multi-tenancy
- Require various deploy methods
- Require S3 storage solution
- Require infrastructure scalability for the future
(for both application and storage)
- Require user-friendly interface for both tenants & admins

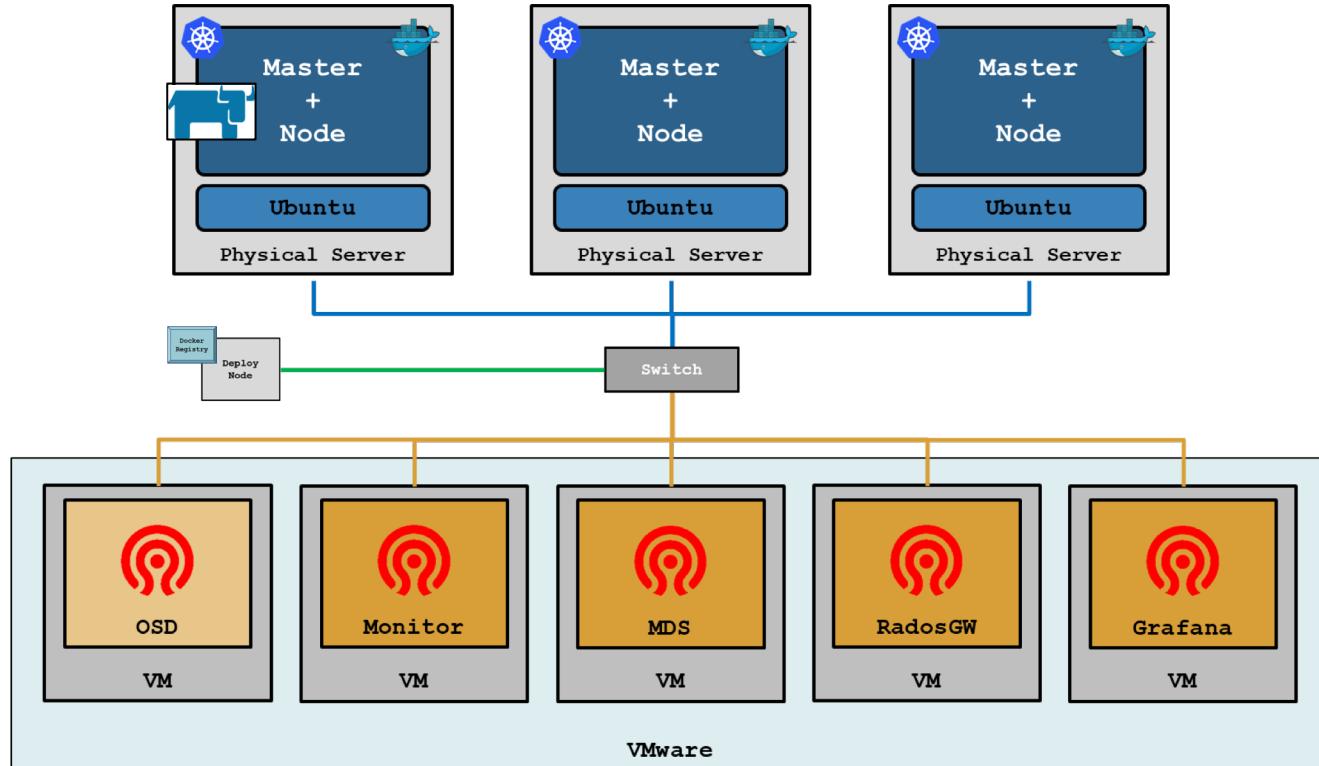


The Challenge

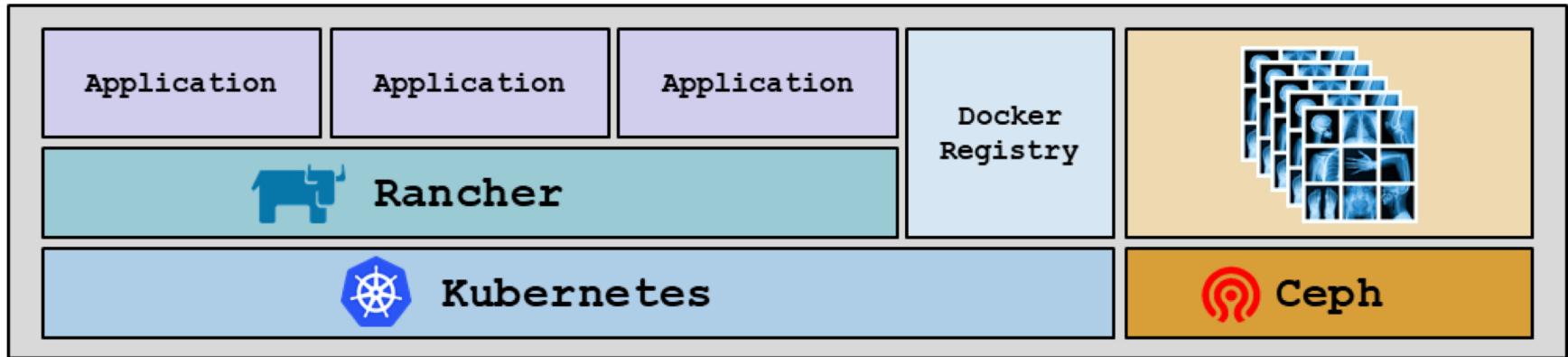
- Docker Swarm is not enough
- Require auto-scaling
- Require multi-tenancy
- Require various deploy methods
- Require S3 storage solution
- Require infrastructure scalability for the future
(for both application and storage)
- Require user-friendly interface for both tenants & admins



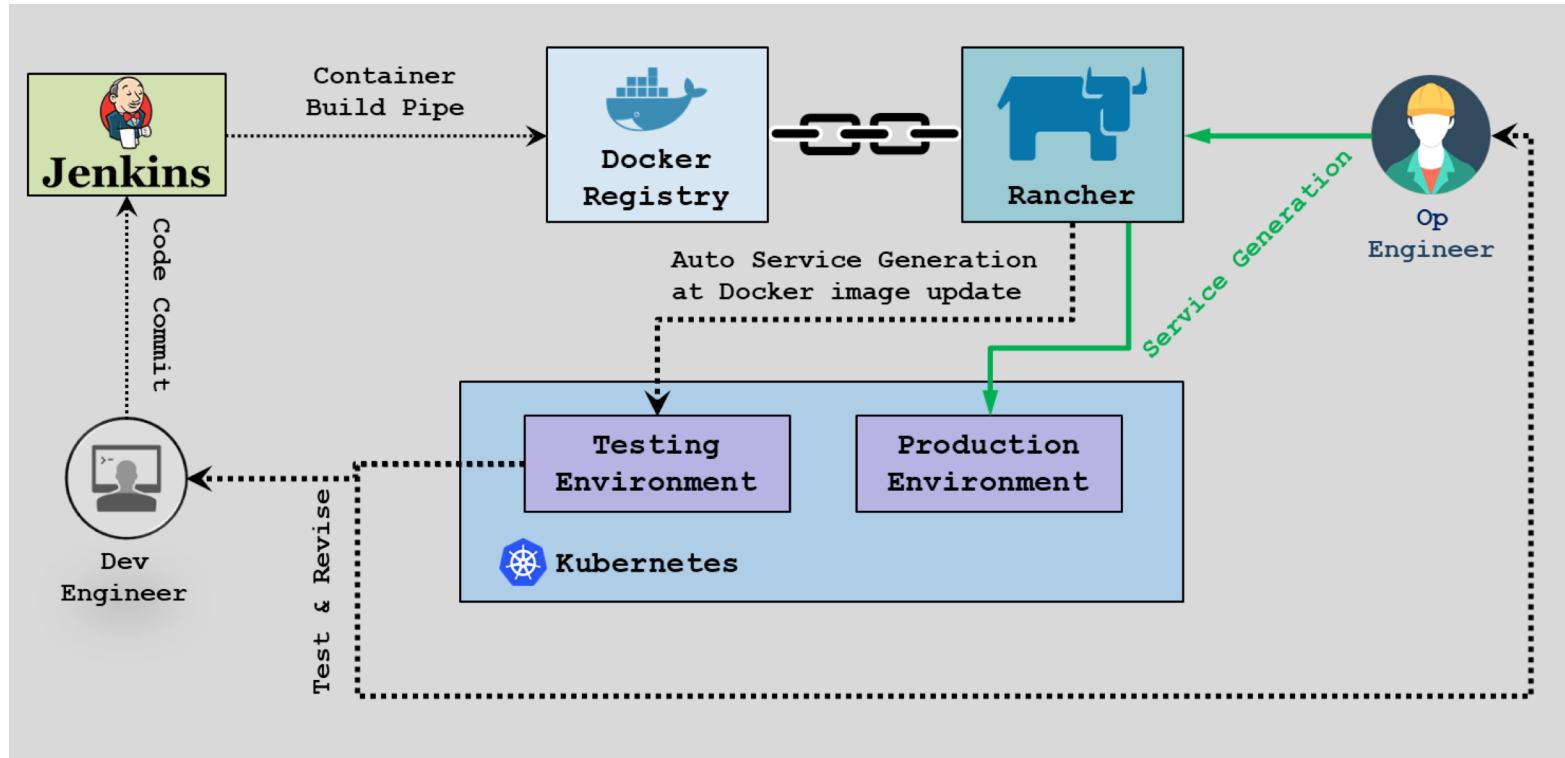
The Solution



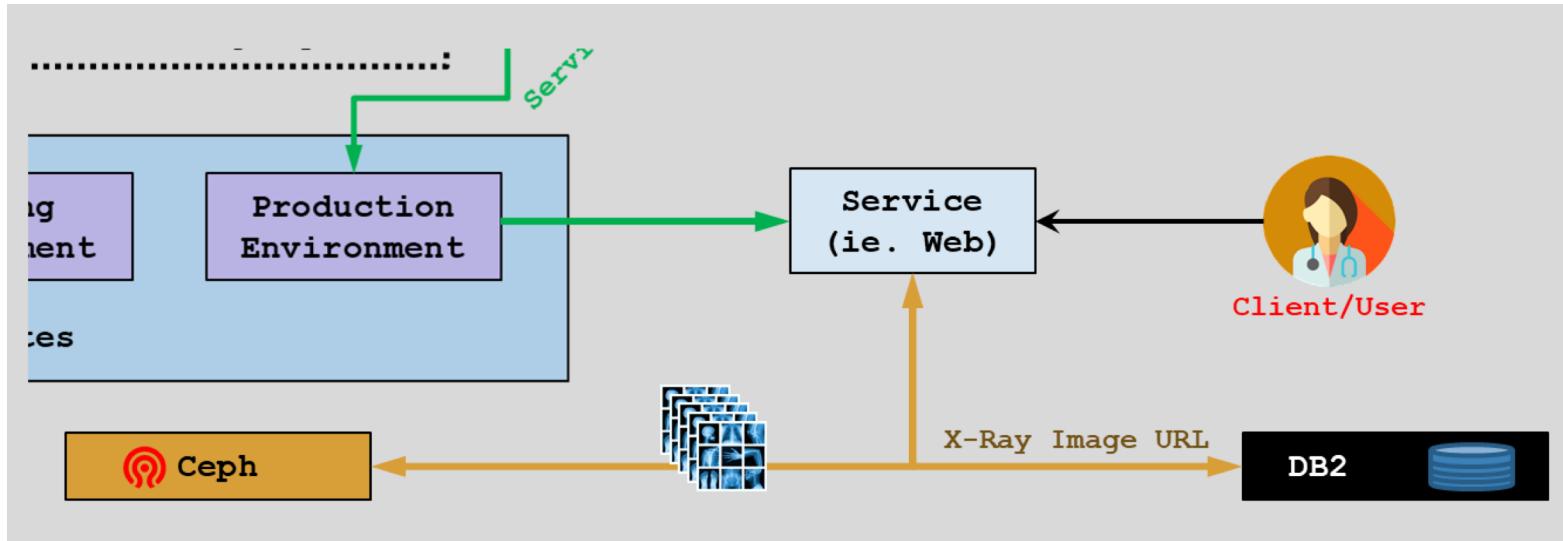
The Solution



The Impact - CI/CD Pipeline



The Impact - Storage Optimization



Conclusion



What should you **REALLY** take away today?



kubernetes

Things to Consider...

- What's best for your app?
i.e. VMs vs Containers
- What are your customized requirements?
i.e. Dev? Ops? DevOps?
- What specific open source apps are in interest?
i.e. Web (wordpress, apache, php);
DB (redis, postgresql, mongodb); ...

Things to Consider...

Determine how much hardware resource is available

Determine what architecture utilizes the most of your resource
(keep application priorities in mind)



ASK inwinSTACK!!



The End



迎棧科技股份有限公司
www.inwinstack.com

The information contained in this document is confidential, privileged and only for the intended recipient and may not be used, published or redistributed without the prior written consent of inwinSTACK Inc.

