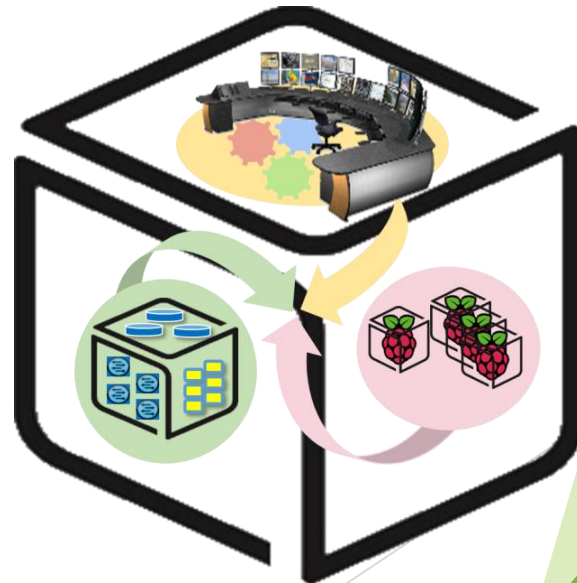


SmartX Labs for Computer Systems

Functions Lab
v1.4
(2018, Spring)
NetCS Lab



History and Contributor of Functions Lab

(2016. 06. 01)

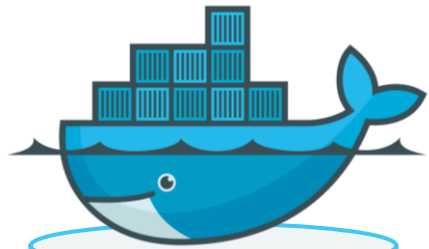
Version	Updated Date	Updated Contents	Contributor
v1.0	20160517	초고 완성	배 정 주
v1.1	20160601	p20, p32 wordpress url을 yourip -> localhost로 변경 (햇갈림 방지)	배 정 주
v1.2	20170510	Docker image 버전 업데이트 및 스크린샷 수정	권 진 철
V1.3	20180124	Container 내부 업데이트 및 iputils-ping패키지 다운	이 승 형
V1.4	20180522	Kubernetes 실습 추가	권 진 철

Functions Lab: Goal

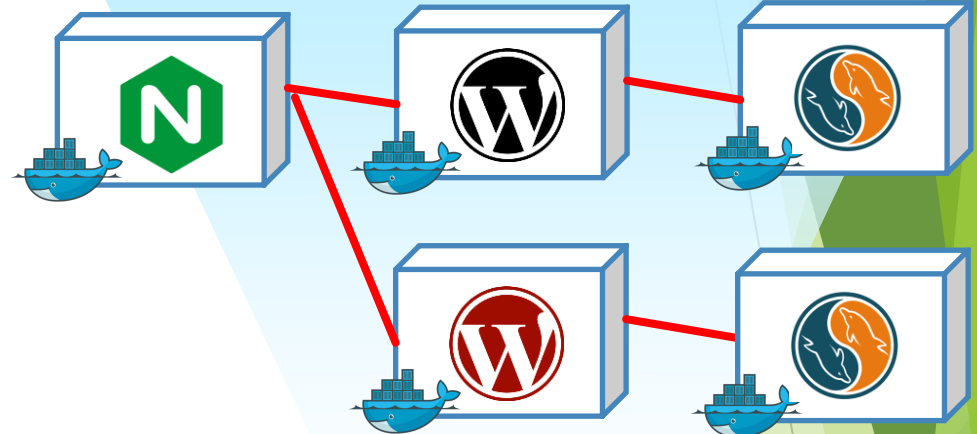
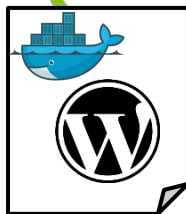
- Strength of Docker: Docker Image
- Introduce Docker Hub
+ Searching and Getting Image from Docker Hub
- Running 3 Tier (nginx-wordpress-mysql) Web Application
- Understanding Docker basic network
(--link option)

Functions Lab: Overall

- One of 3-Tier example



Docker Hub



Functions Lab: Overall

- Background knowledge of goal

We will running one of web application: Wordpress

This web application is consisted with 3 containers: nginx, wordpress, mysql



nginx

: A http server which has following features.

- **Reverse proxying**
- SSL TLS SNI support... and etc

Usually, It is compared by Apache.



wordpress

: It is web software to create website, blog, or application.

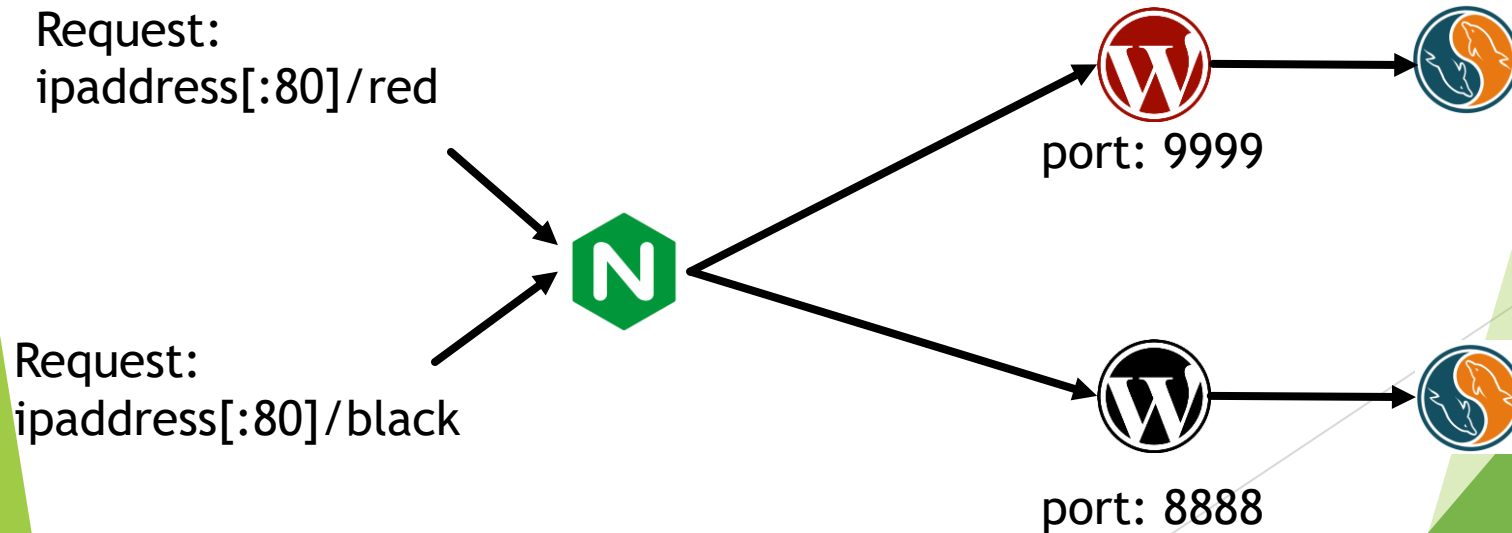
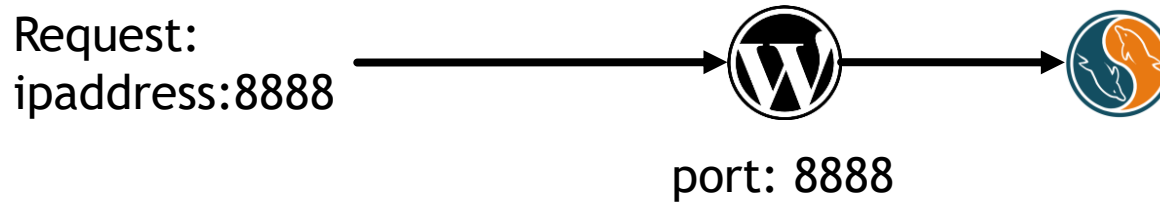


mysql

: Relational Database Management System(RDBMS)

Functions Lab: Overall

- Scenario



Prerequisite for Functions lab

Functions lab focus on (Docker container) functions of NUC



docker-engine (version: 1.10 or above)

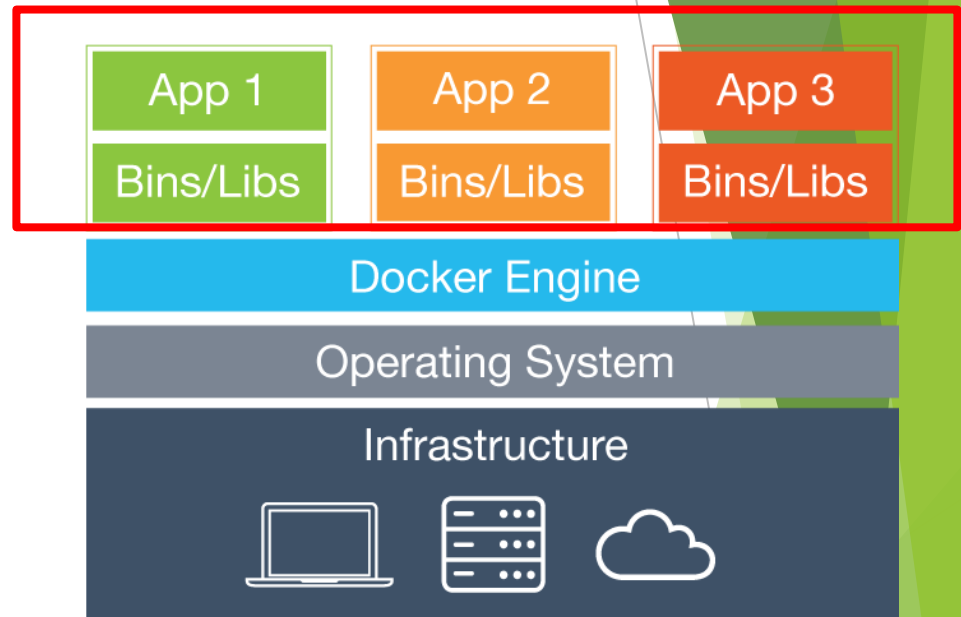
Docker Background Knowledge

- Reminder: Docker Image

Docker image: A file which contains status of Docker container.
Similar to snapshot of VM.

It can be branched and versioned...

Docker image can be shared easily.
(It is important feature of Docker.)

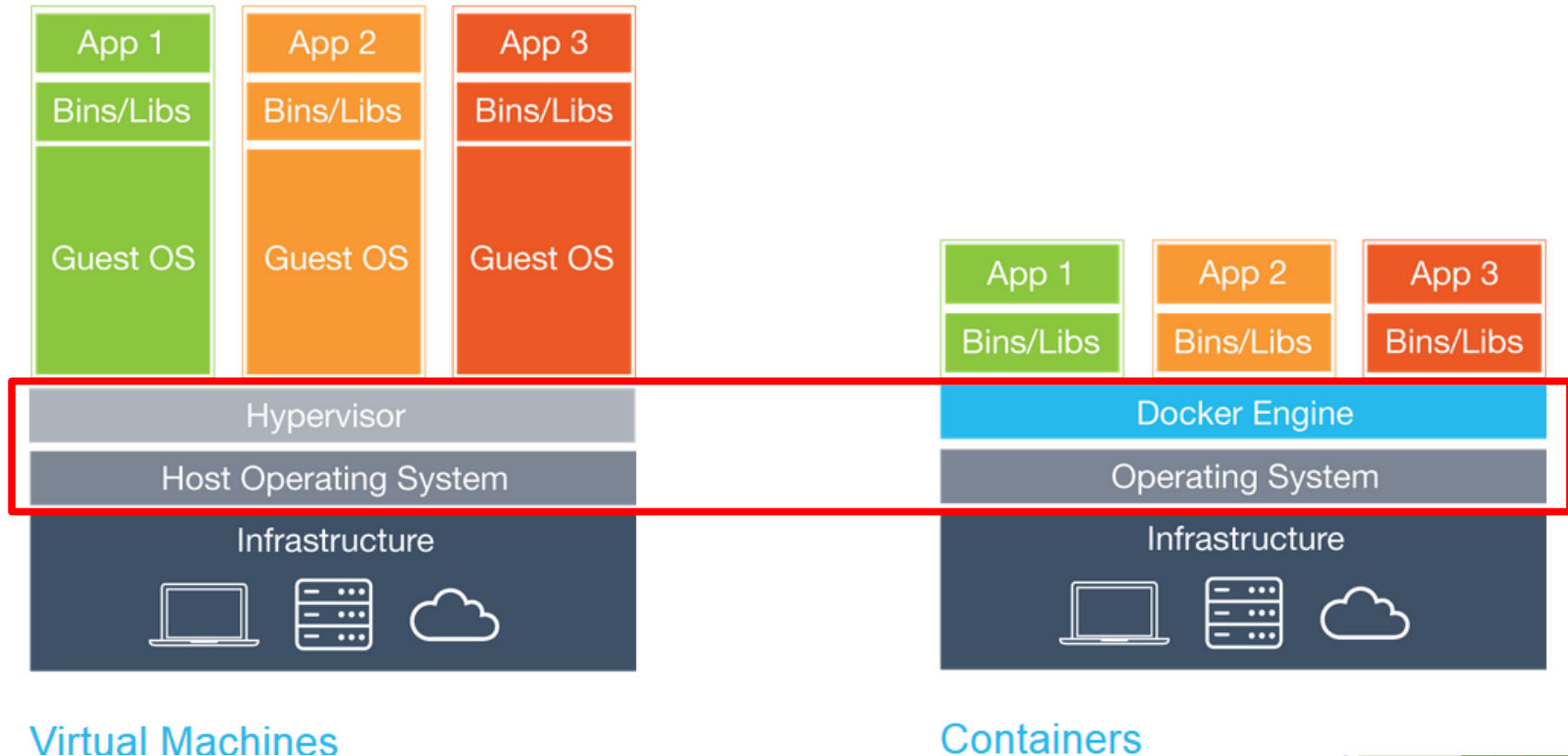


You can find Ubuntu image on your machine

```
bjj@ . :~$ docker images
REPOSITORY              TAG                IMAGE ID
CENTROSOFT/centos        latest             778a53015523
CENTROSOFT/centos        4 weeks ago       196.7 MB
CENTROSOFT/ubuntu        latest             14b59d36bae0
CENTROSOFT/ubuntu        10 weeks ago      187.9 MB
```


Docker Background Knowledge

- Reminder: Container restraint



Since container uses host kernel, **OS** of host should be **Linux** distribution.

Docker Background Knowledge

- Why Docker image can be shared easily? (1)

Commands:

attach	Attach 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
pull	Pull an image or a repository from a registry
push	Push an image or a repository to a registry

Docker provides related commands!

This usage is similar to code management system. (e.g, svn, git)

Docker Background Knowledge

- Why Docker image can be shared easily? (2)

```
FROM resin/rpi-raspbian:wheezy
MAINTAINER Seungryong Kim <srkim@nm.gist.ac.kr>
```

```
#Update & Install wget, vim
RUN apt-get update
RUN apt-get -y install wget
RUN apt-get -y install vim
```

```
#Timezone
RUN cp /usr/share/zoneinfo/Asia/Seoul /etc/localtime
```

```
#Install Oracle JAVA
RUN mkdir -p /opt
RUN wget --no-cookies --no-check-certificate --header "Cookie: gpw_e24=http%3A%2F%2Fwww.oracle.com%2F" http://www.oracle.com/technetwork/java/javase-downloads-138463.zip -O /opt/jdk1.8.0_33.zip
```

```
#Configure environmental variables
ENV JAVA_HOME /opt/jdk1.8.0_33
ENV PATH $PATH:/opt/jdk1.8.0_33/bin
RUN ln -s /opt/jdk1.8.0_33/bin/java /usr/bin/java
```

```
#Install Flume
RUN sudo wget --no-check-certificate http://www.apache.org/dist/flume/1.6.0/flume-1.6.0-bin.tar.gz -O /tmp/flume-1.6.0-bin.tar.gz
RUN sudo mv apache-flume-1.6.0-bin /flume
```

```
ADD plugins.d /flume/plugins.d
ADD flume-conf.properties /flume/conf/
```

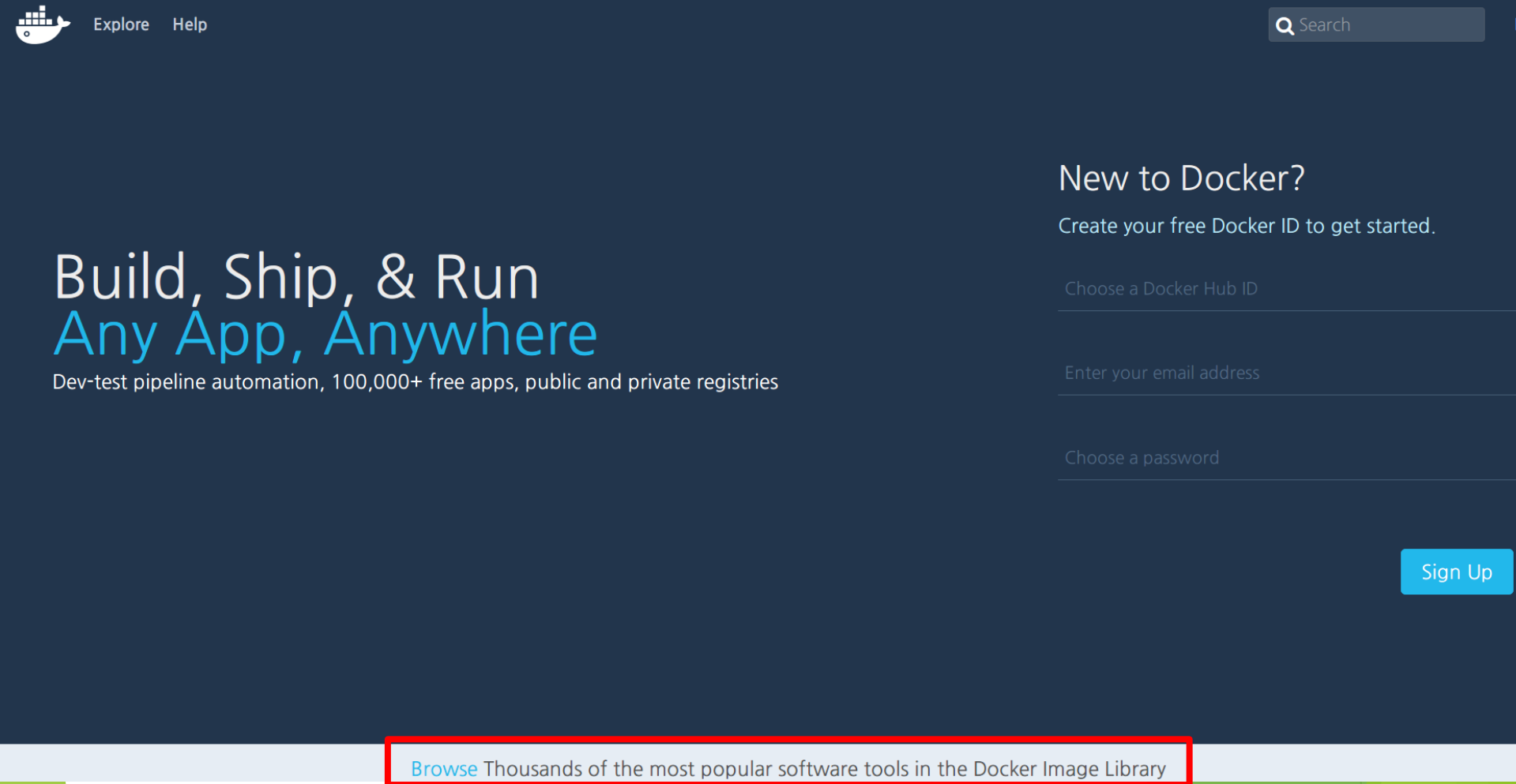
```
#Working directory
WORKDIR /flume
```

Docker image is built by Dockerfile which is small text file.
That means sharing Docker image does not requires huge bandwidth (sometimes).

Getting Docker Image

- Public Docker Image Repository: Docker Hub (1)

<https://hub.docker.com/>



The screenshot shows the Docker Hub homepage. At the top, there is a dark blue header with the Docker logo (a white ship icon) on the left, and 'Explore' and 'Help' links in the center. On the right side of the header is a search bar with a magnifying glass icon and the text 'Search'. Below the header, the main content area has a dark blue background. On the left, the text 'Build, Ship, & Run Any App, Anywhere' is displayed in white and light blue, with 'Any App, Anywhere' in a larger, light blue font. Below this, a subtitle reads 'Dev-test pipeline automation, 100,000+ free apps, public and private registries'. On the right side, there is a section titled 'New to Docker?' in white, followed by the text 'Create your free Docker ID to get started.' Below this are three input fields: 'Choose a Docker Hub ID', 'Enter your email address', and 'Choose a password'. A blue 'Sign Up' button is located at the bottom right of this section. At the very bottom of the page, a light blue banner contains the text 'Browse Thousands of the most popular software tools in the Docker Image Library', which is highlighted with a red rectangular border.

Build, Ship, & Run
Any App, Anywhere

Dev-test pipeline automation, 100,000+ free apps, public and private registries

New to Docker?

Create your free Docker ID to get started.

Choose a Docker Hub ID

Enter your email address

Choose a password

Sign Up

[Browse](#) Thousands of the most popular software tools in the Docker Image Library

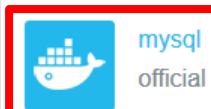
Getting Docker Image

- Public Docker Image Repository: Docker Hub (2)

You can easily find application image



Repositories (3268)



2.2K
STARS

10M+
PULLS



Repositories (1028)



950
STARS

5M+
PULLS

>
DETAILS

Getting Docker Image

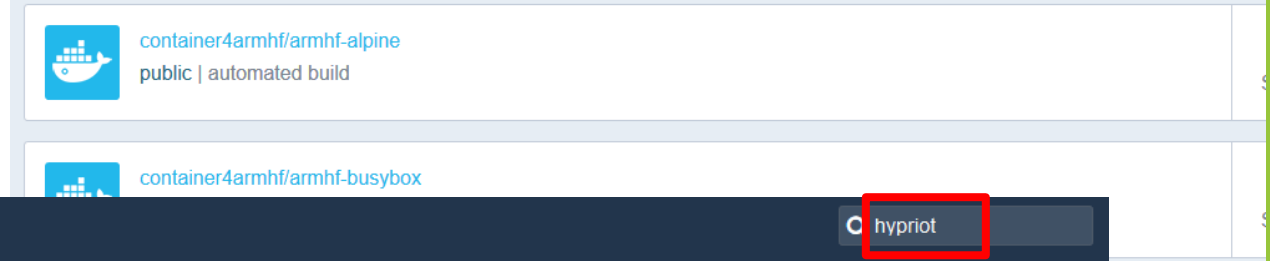
- Be aware!



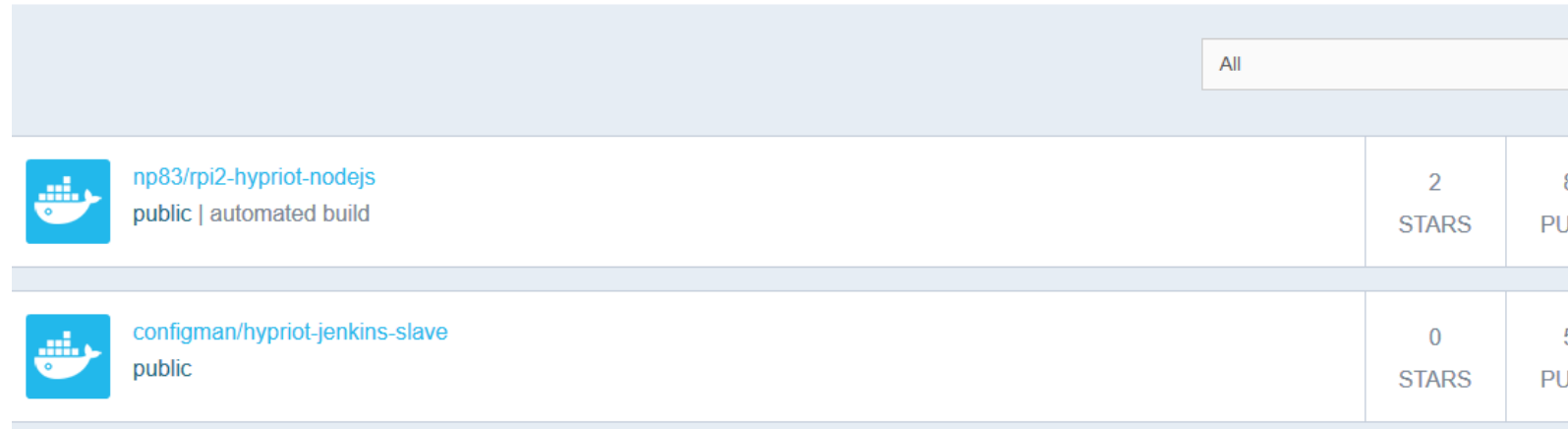
Repositories (2360)

x86 based image can not run in Raspberry Pi!

If you want to use Docker image in RPi, you should find ARM based image!



Repositories (41)



Getting Docker Image

- Public Docker Image Repository: Docker Hub (3)



Before using Docker image, **you should read** description.
It is very important because required option rely on image.

OFFICIAL REPOSITORY

mysql ☆

Last pushed: 2 days ago

Repo Info Tags

Short Description

MySQL is a widely used, open-source relational database management system (RDBMS).

Full Description

Supported tags and respective


Starting a MySQL instance is simple:

```
$ docker run --name some-mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:tag
```

Getting Docker Image

- Public Docker Image Repository: Docker Hub (4)

OFFICIAL REPOSITORY

wordpress 

Last pushed: 2 days ago

Repo Info [Tags](#)

Short Description

The WordPress rich content management system can utilize plugins, widgets, and themes.

Full Description

Supported tags and respective

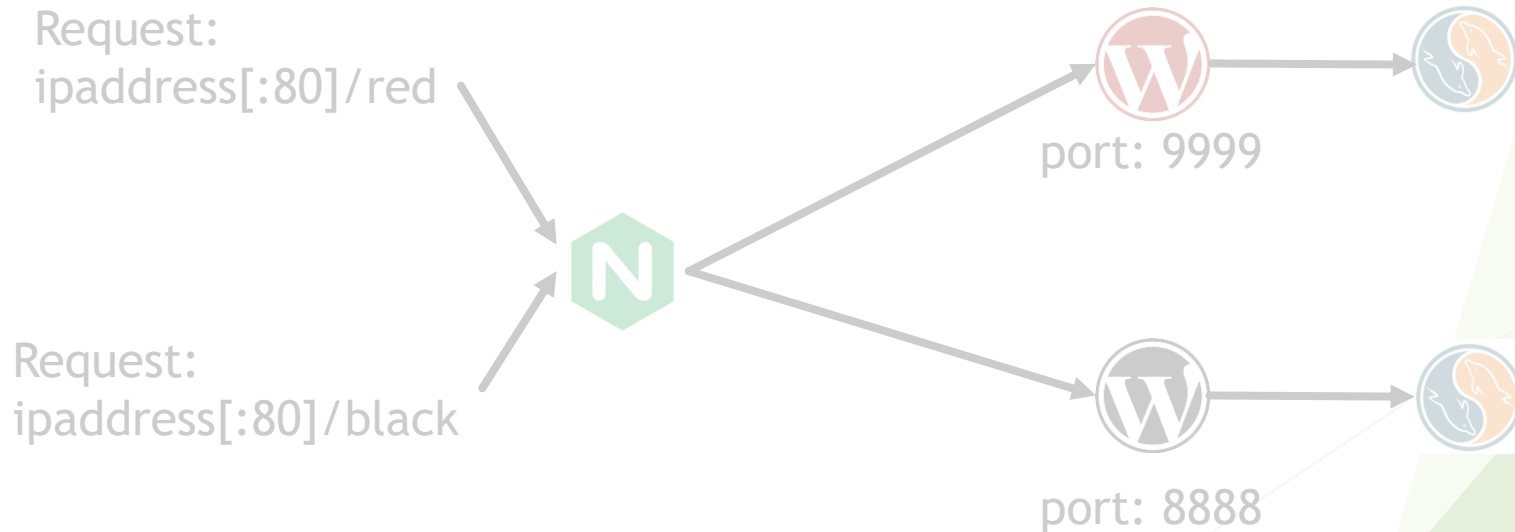
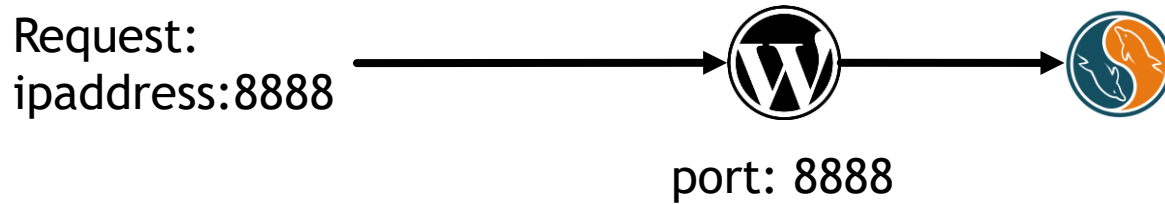


How to use this image

```
$ docker run --name some-wordpress --link some-mysql:mysql -d wordpress
```


Functions Lab:

- Try: wordpress-mysql



Running Web Application

- Run mysql container



```
mkdir ~/sql
```

```
sudo docker run --name word_sql -v /home/[username]/sql:/var/lib/mysql -e  
MYSQL_ROOT_PASSWORD=[password] -d mysql:5.7.18
```

Tag. Default is latest (Latest version)

```
tein@vbox-develop:~$ docker run --name word_sql -v /home/tein/sql:/var/li  
b/mysql -e MYSQL_ROOT_PASSWORD=functions -d mysql:5.7.18  
5cba5d67f49c412ee477c8e803795d06fd477758f150678bfd526a9aeed62d7b
```

```
sudo docker ps
```

```
tein@vbox-develop:~$ docker ps
```

CONTAINER ID	STATUS	IMAGE	PORTS	COMMAND	NAMES	CREATED
5cba5d67f49c	Up 2 seconds	mysql:5.7.18	3306/tcp	"docker-entrypoint.sh"	word_sql	3 seconds ago

Running Web Application

- Run wordpress container



```
sudo docker run --name wordpress --link word_sql:mysql -p 8888:80 -d  
wordpress:4.7.4-apache
```

(Will be introduced)

```
tein@vbox-develop:~$ docker run --name wordpress --link word_sql:mysql -p  
8888:80 -d wordpress:4.7.4-apache  
752dcdfdd3881a78b1bdee2ead1ac81837730289ac511448c19d73c696a0d743
```

```
sudo docker ps
```

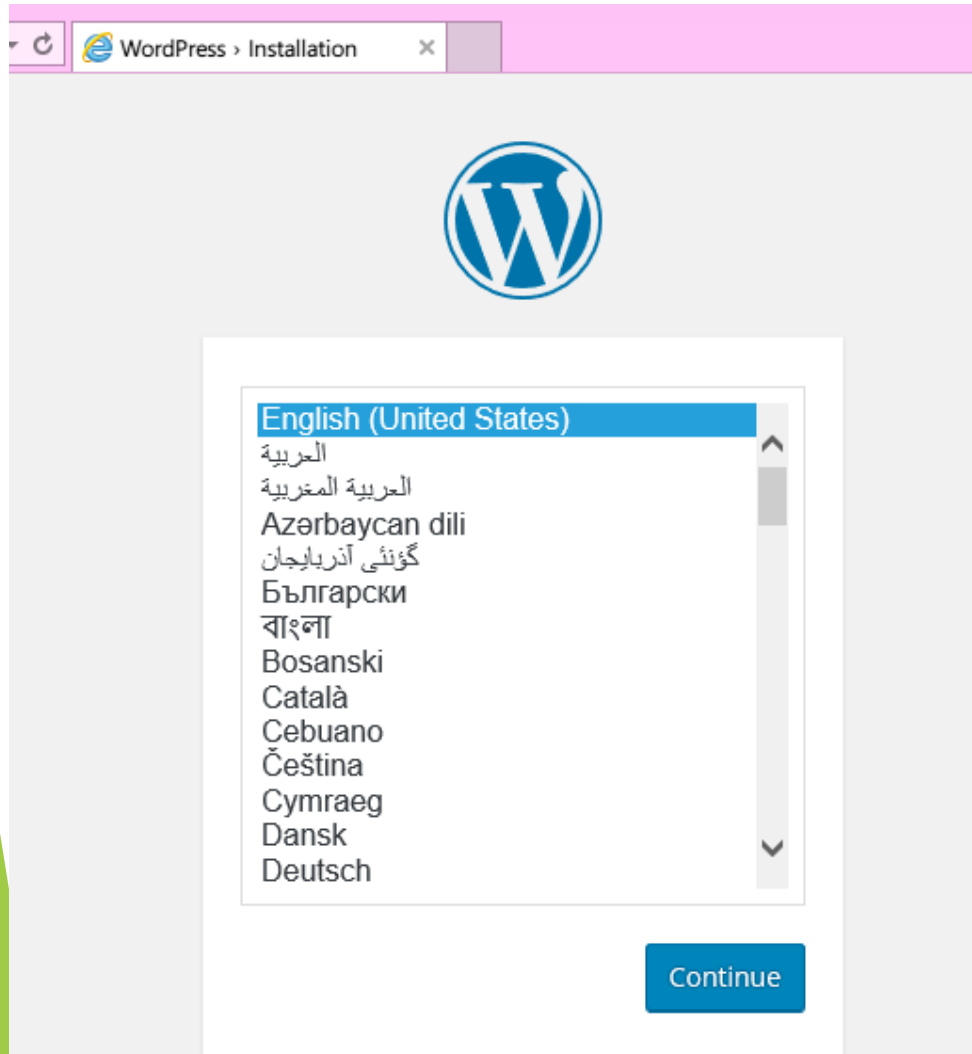
```
tein@vbox-develop:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
752dcdfdd388	wordpress:4.7.4-apache	"/entrypoint.sh apach"	3 seconds ago	Up 2 seconds	0.0.0.0:8888->80/tcp	wordpress
5cba5d67f49c	mysql:5.7.18	"docker-entrypoint.sh"	5 minutes ago	Up 5 minutes	3306/tcp	word_sql

Running Web Application

- Check wordpress!

<http://localhost:8888>



We typed just 2 line of commands to running wordpress!

This is one of major Docker strength :Easy deployment of software

Running Web Application

- Default configuration: Wordpress

The WordPress language selection screen. At the top is the WordPress logo. Below it is a scrollable list of languages. The list includes: Gàidhlig, Galego, هزاره گی, עברית, हिन्दी, Hrvatski, Magyar, Հայերեն, Bahasa Indonesia, Íslenska, Italiano, 日本語, ქართული, and 한국어. The '한국어' option is highlighted in blue. At the bottom right of the list is a blue button labeled '계속' (Continue).

black

다른 워드프레스 사이트

안녕하세요!

The WordPress installation configuration screen. At the top is the WordPress logo. Below it is the heading '환영합니다' (Welcome). A paragraph of text follows: '유명한 5분 워드프레스 설치 과정에 오신 것을 환영합니다! 아래에서 정보를 입력만 하면 세계에서 가장 확장성 높고 강력한 개인 출판 플랫폼을 사용하는 길로 들어서게 됩니다.' Below this is the heading '필요한 정보' (Required information). A paragraph of text follows: '다음 정보들을 제공해주세요. 나중에 다시 변경할 수 있으니 걱정하지 않아도 됩니다.' There are four form fields: '사이트 제목' (Site title) with the value 'black', '사용자명' (Username) with the value 'cslab', '비밀번호' (Password) with the value 'function' and a '숨기기' (Hide) button, and '비밀번호 확인' (Confirm password) with a checked box for '약한 패스워드 사용 확인' (Check weak password usage). Below these is the '이메일 주소' (Email address) field with the value 'cslab@functions.com'. A final paragraph of text at the bottom says: '계속하기 전에 이메일 주소를 한 번 더 확인하세요.'

Running Web Application

- About --link option (1)

(In official docs..)

Docker also has a linking system that allows you to **link multiple containers together** and **send connection information from one to another**. When containers are linked, **information about a source container can be sent to a recipient container**. This allows the recipient to see selected data describing aspects of the source container.

Links allow containers to discover each other and securely transfer information about one container to another container. When you set up a link, you create a conduit between a source container and a recipient container.

Usage: `--link <name or id>:alias`
`--link <name or id>`

Naming container is important!

Running Web Application

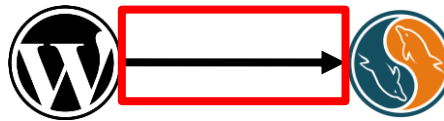
- About --link option (2)



```
sudo docker inspect -f "{{ .HostConfig.Links }}" wordpress
```

```
tein@vbox-develop:~$ docker inspect -f "{{ .HostConfig.Links }}" wordpress  
[/word_sql:/wordpress/mysql]
```

You can see wordpress container is linked with mysql container.



Conduit which is made by --link option

Running Web Application

- About --link option (3)

(In official docs..)

Docker **creates a secure tunnel** between the containers that **doesn't need to expose any ports externally on the container.**

That's a **big benefit** of linking: **we don't need to expose the source container.**

Review our commands: Creating mysql container
→ This container doesn't expose any ports.

```
mkdir sql  
sudo docker run --name word_sql -v /home/[username]/sql:/var/lib/mysql -e  
MYSQL_ROOT_PASSWORD=[password] -d mysql
```

Also we don't need to type password of mysql when creating wordpress container

```
sudo docker run --name wordpress --link word_sql:mysql -p 80:80 -d  
wordpress
```


Running Web Application

- About --link option (4)

(In official docs..)

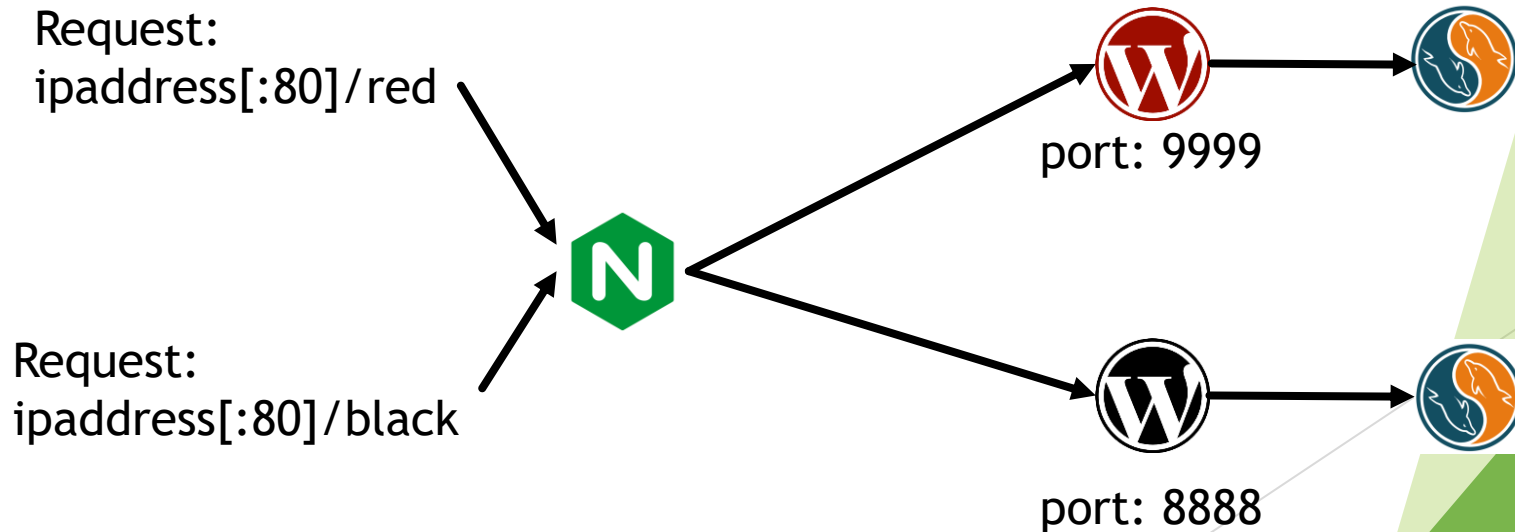
Functionality of this option is:

- Updating Environment variables
- Updating the /etc/hosts file

```
tein@vbox-develop:~$ docker run -it --name=container1 ubuntu /bin/bash
root@af6fa8caa3ff:/# cat /etc/hosts
127.0.0.1      localhost
::1           localhost ip6-localhost ip6-loopback
fe00::0       ip6-localnet
ff00::0       ip6-mcastprefix
ff02::1       ip6-allnodes
ff02::2       ip6-allrouters
172.17.0.6     c 2dd9a5045883 container1
172.17.0.7     af6fa8caa3ff
root@af6fa8caa3ff:/# apt-get update | apt-get install iputils-ping
Reading package lists... Done
Building dependency tree
Reading state information... Done
iputils-ping is already the newest version (3:20121221-5ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.
root@af6fa8caa3ff:/# ping c
PING c (172.17.0.6) 56(84) bytes of data.
64 bytes from c (172.17.0.6): icmp_seq=1 ttl=64 time=0.139 ms
64 bytes from c (172.17.0.6): icmp_seq=2 ttl=64 time=0.088 ms
^C
--- c ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1007ms
rtt min/avg/max/mdev = 0.088/0.113/0.139/0.027 ms
root@af6fa8caa3ff:/# ping container1
PING c (172.17.0.6) 56(84) bytes of data.
64 bytes from c (172.17.0.6): icmp_seq=1 ttl=64 time=0.151 ms
64 bytes from c (172.17.0.6): icmp_seq=2 ttl=64 time=0.092 ms
^C
--- c ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1031ms
rtt min/avg/max/mdev = 0.092/0.121/0.151/0.031 ms
```

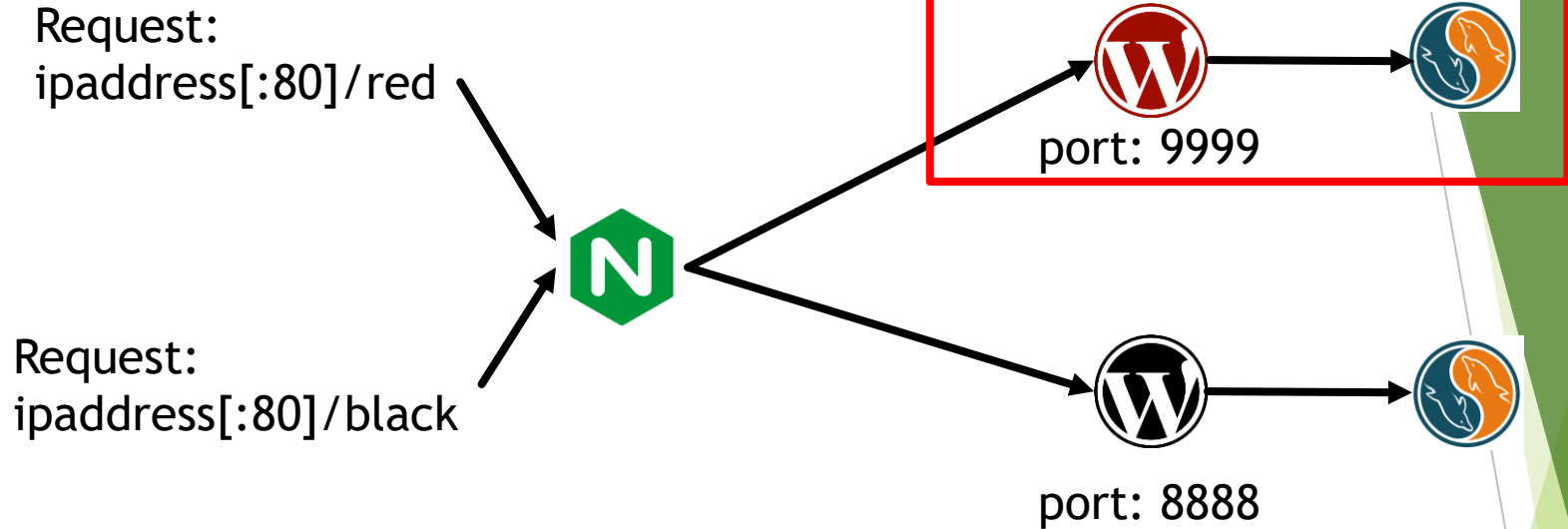
Functions Lab:

- Try: nginx-wordpress-mysql



Running Web Application

- Make another wordpress container set



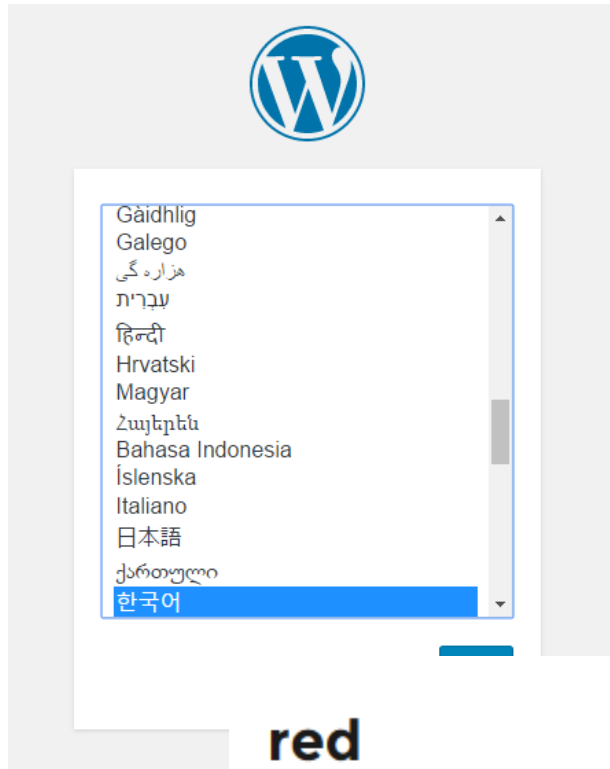
```
mkdir ~/sql2
```

```
sudo docker run --name word_sql2 -v /home/[username]/sql2:/var/lib/mysql -e  
MYSQL_ROOT_PASSWORD=[password] -d mysql:5.7.18
```

```
sudo docker run --name wordpress2 --link word_sql2:mysql -p 9999:80 -d  
wordpress:4.7.4-apache
```

Running Web Application

- Default configuration: Wordpress



다른 워드프레스 사이트

A screenshot of the WordPress installation '환영합니다' (Welcome) screen. The WordPress logo is at the top. The text '환영합니다' is followed by a paragraph: '유명한 5분 워드프레스 설치 과정에 오신 것을 환영합니다! 아래에서 정보를 입력만 하면 세계에서 가장 확장성 높고 강력한 개인 출판 플랫폼을 사용하는 길로 들어서게 됩니다.' Below this is the '필요한 정보' (Required information) section. It includes fields for '사이트 제목' (Site title) with the value 'red', '사용자명' (Username) with the value 'cslab', and '비밀번호' (Password) with the value 'function'. There is a '숨기기' (Hide) button next to the password field. Below the password field is a red button labeled '매우 약함' (Very weak). A note says '중요: 로그인하려면 이 비밀번호가 필요할 것입니다. 안전한 곳에 보관하십시오.' (Important: You will need this password to log in. Please keep it in a safe place.) There is a checkbox for '비밀번호 확인' (Confirm password) which is checked, with the text '약한 패스워드 사용 확인' (Confirm weak password use). Below this is the '이메일 주소:' (Email address) field with the value 'cslab@functions.com'. A note says '계속하기 전에 이메일 주소를 한 번 더 확인하세요.' (Please check your email address one more time before continuing.) There is a checkbox for '검색 엔진 접근 여부' (Search engine access) which is unchecked, with the text '검색 엔진이 이 사이트 검색 차단하기' (Prevent search engines from indexing this site) and a note '이 요청이 받아들여지는 것은 전적으로 검색 엔진에 좌우됩니다.' (Whether this request is accepted is entirely up to the search engine.) At the bottom is a button labeled '워드프레스 설치하기' (Install WordPress).

안녕하세요!

Running Web Application

- Current status

black

다른 워드프레스 사이트

ip:8888

안녕하세요!

red

다른 워드프레스 사이트

ip:9999

안녕하세요!

Running Web Application

- Make container for reverse proxy



```
sudo docker run -it --name=nginx -p 80:80 ubuntu:16.04
```

Forwarding host_port:container_port

Running Web Application

- Setting reverse proxy



```
sudo apt-get update  
sudo apt-get install nginx  
sudo apt-get install vim
```

```
cd /etc/nginx/sites-enabled  
rm default  
vi default
```

```
server {  
    listen 80;  
    location /black {  
        proxy_pass http:// [your ip] :8888/  
    }  
    location /red {  
        proxy_pass http:// [your ip] :9999/  
    }  
}
```

```
service nginx start
```

Running Web Application

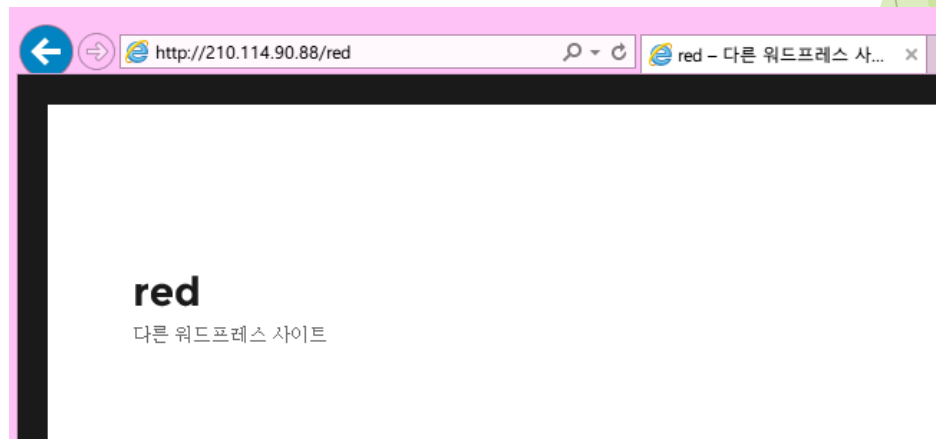
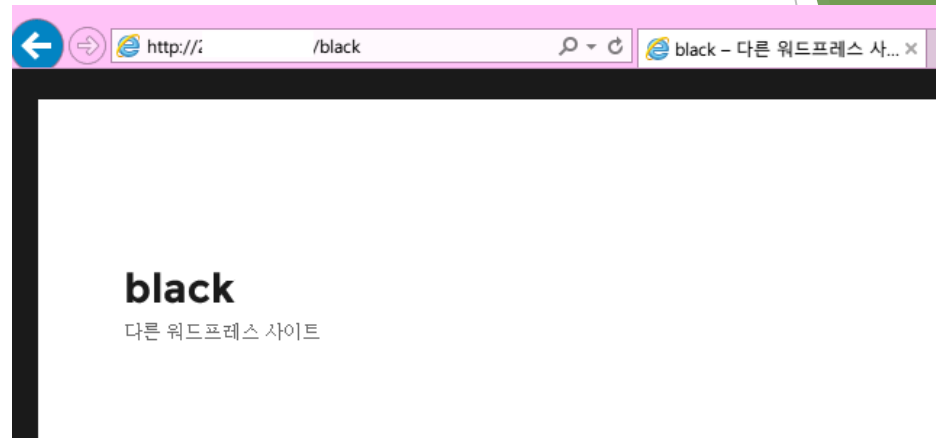
- Setting reverse proxy



Try

`http://localhost/black`

`http://localhost/red`



Running Web Application

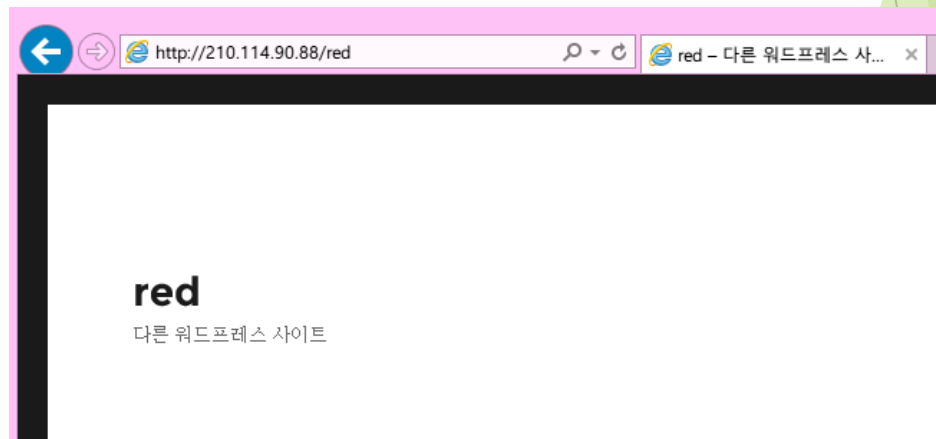
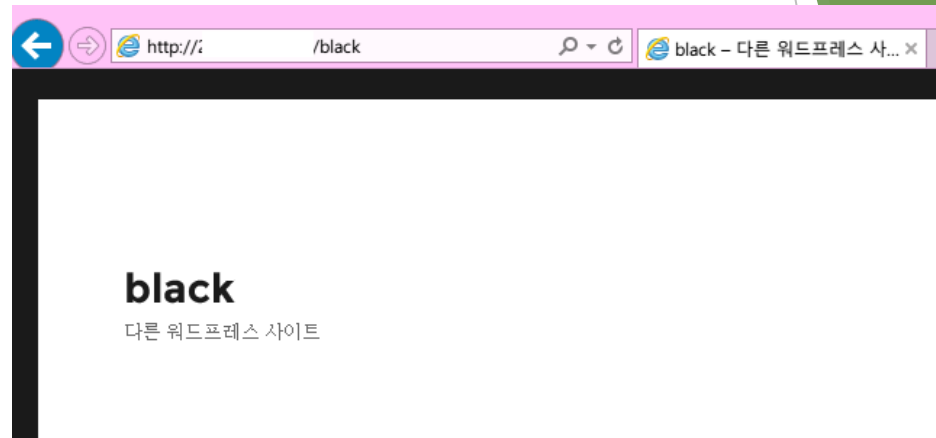
- Setting reverse proxy



Try

`http://localhost/black`

`http://localhost/red`



Running Web Application

- About Kubernetes (1)



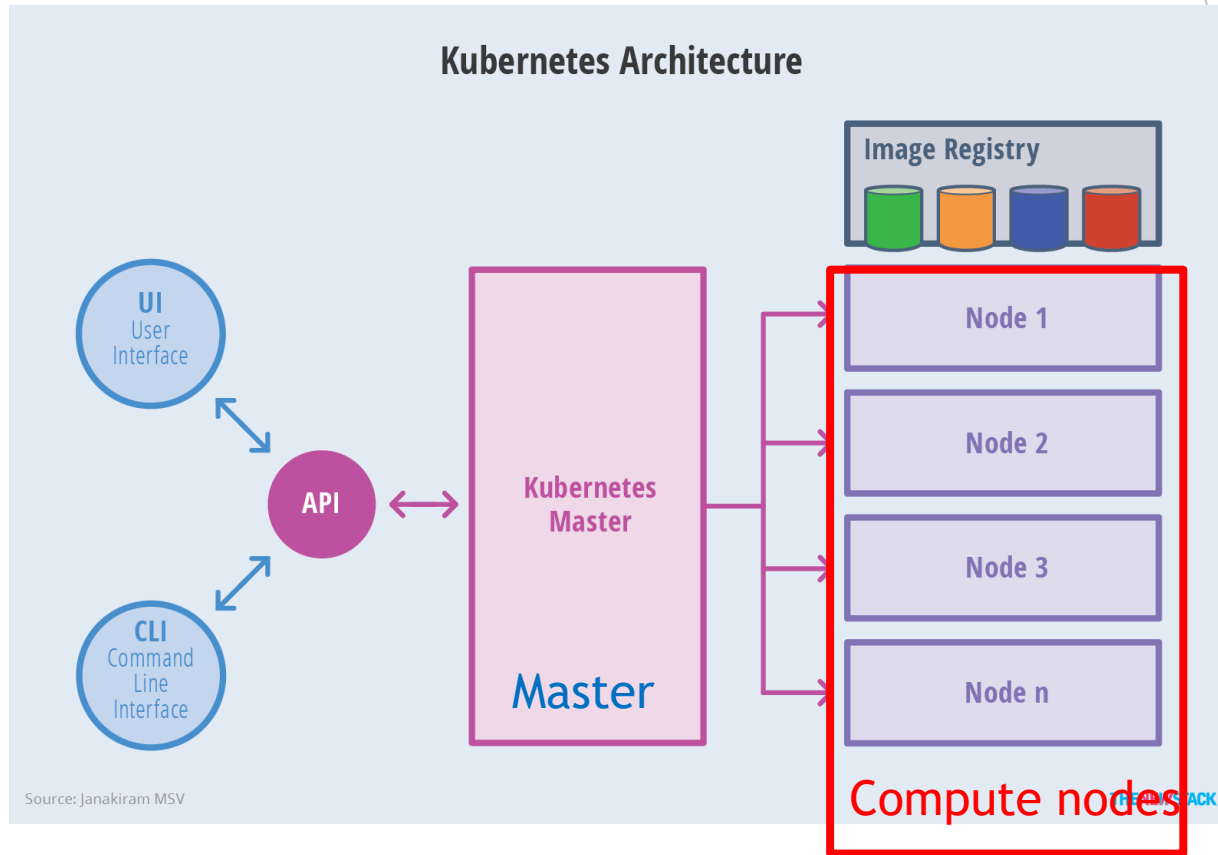
Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

Kubernetes Features

- **Horizontal scaling:** Scale your application up and down with a simple command, with a UI, or automatically based on CPU usage.
- **Self-healing:** Restarts containers that fail, replaces and reschedules containers when nodes die, kills containers that don't respond to your user-defined health check, and doesn't advertise them to clients until they are ready to serve.
- **Service discovery and load balancing:** No need to modify your application to use an unfamiliar service discovery mechanism. Kubernetes gives containers their own IP addresses and a single DNS name for a set of containers, and can load-balance across them.
- **Storage Orchestration:** Automatically mount the storage system of your choice, whether from local storage, a public cloud provider

Running Web Application

- About Kubernetes (2)

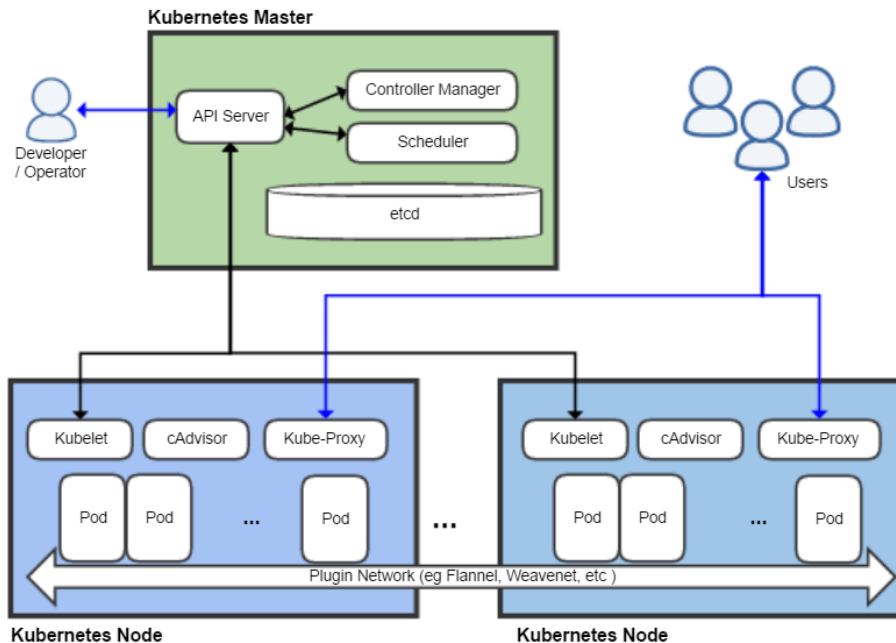


Kubernetes cluster consists of **at least one master** and multiple **compute nodes**.

- **Computing Cluster** is a form of computing in which a group of computers are linked together so that they can act like a single entity
→ You will be learn more in the next lab.

Running Web Application

- About Kubernetes (3)



The **master** is responsible for exposing the application program interface (**API**), scheduling the deployments and managing the overall cluster.

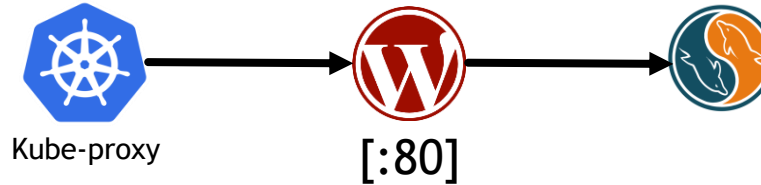
Pod is consists of one or more containers that are guaranteed to be co-located on the host machine and can share resources. Each pod is assigned a unique IP address within the cluster, which allows applications to use ports without the risk of conflict.

Running Web Application

- Make a wordpress container set With Kubernetes



Request:
ipaddress[:30080]

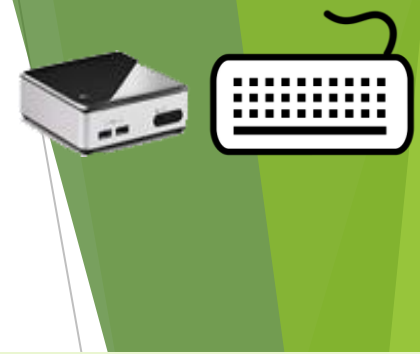


Install kubernetes

```
sudo su
apt-get update && apt-get install -y apt-transport-https curl
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
>deb http://apt.kubernetes.io/ kubernetes-xenial main
>EOF
apt-get update
apt-get install -y kubelet kubeadm kubectl
```

Running Web Application

- Configure Kubernetes (1)



Edit kubernetes configure file

```
sudo vi /etc/systemd/system/kubelet.service.d/10-kubeadm.conf  
→ Add [ Environment="KUBELET_EXTRA_ARGS=--fail-swap-on=false" ]  
sudo systemctl daemon-reload  
sudo systemctl restart kubelet
```

```
[Service]  
Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf"  
Environment="KUBELET_SYSTEM_PODS_ARGS=--pod-manifest-path=/etc/kubernetes/manifests --allow-privileged=true"  
Environment="KUBELET_NETWORK_ARGS=--network-plugin=cni --cni-conf-dir=/etc/cni/net.d --cni-bin-dir=/opt/cni/bin"  
Environment="KUBELET_DNS_ARGS=--cluster-dns=10.96.0.10 --cluster-domain=cluster.local"  
Environment="KUBELET_AUTHZ_ARGS=--authorization-mode=Webhook --client-ca-file=/etc/kubernetes/pki/ca.crt"  
Environment="KUBELET_CADVISOR_ARGS=--cadvisor-port=0"  
Environment="KUBELET_CERTIFICATE_ARGS=--rotate-certificates=true --cert-dir=/var/lib/kubelet/pki"  
Environment="KUBELET_EXTRA_ARGS=--fail-swap-on=false" Add the line like this  
ExecStart=  
ExecStart=/usr/bin/kubelet $KUBELET_KUBECONFIG_ARGS $KUBELET_SYSTEM_PODS_ARGS $KUBELET_NETWORK_ARGS $KUBELET_DNS_ARGS $KUBELET_AUTHZ_ARGS $KUBELET_CADVISOR_ARGS $KUBELET_CERTIFICATE_ARGS $KUBELET_EXTRA_ARGS
```

Run a kubernetes cluster: Your NUC will be a Worker node as well as a Master.

```
sudo kubeadm reset  
kubeadm init --skip-preflight-checks
```

Running Web Application

- Configure Kubernetes (2)



Run these commands to make kubectl work **for your non-root user**.
Make sure that you are not in root

```
mkdir -p $HOME/.kube  
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Check your cluster status

```
sudo kubectl get nodes
```

```
root@labs:/home/netcs# kubectl get nodes  
NAME      STATUS    ROLES    AGE   VERSION  
labs      NotReady  master   34m   v1.10.3
```

Your NUC is running as a kubernetes master.

But pods are not allowed to be scheduled to run on the master

It allows pods to be scheduled to run on the Kubernetes master server

```
kubectl taint nodes --all node-role.kubernetes.io/master-
```

Running Web Application

- Configure Kubernetes (3)



You **MUST** install a pod network add-on so that your pods can communicate with each other. We will use Weave in this lab.

Install Weave (pod network add-on)

```
kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"
```

Make sure Weave(network add-on) works

```
sudo kubectl get nodes
```

```
root@labs:/home/netcs# kubectl get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
labs	Ready	master	2h	v1.10.3

```
kubectl get po -n kube-system -o wide
```

```
root@labs:/home/netcs# kubectl get po -n kube-system -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE
etcd-labs	1/1	Running	0	2h	203.237.53.84	labs
kube-apiserver-labs	1/1	Running	0	2h	203.237.53.84	labs
kube-controller-manager-labs	1/1	Running	0	2h	203.237.53.84	labs
kube-dns-86f4d74b45-8q9tg	3/3	Running	0	2h	10.32.0.2	labs
kube-proxy-z2qqx	1/1	Running	0	2h	203.237.53.84	labs
kube-scheduler-labs	1/1	Running	0	2h	203.237.53.84	labs
weave-net-nhf4n	2/2	Running	0	2m	203.237.53.84	labs

Running Web Application

- Run Wordpress application (1)



We will run Wordpress application using kubernetes template which describes how to run wordpress and sql container.

Get the yaml template

```
wget -O wordpress.yaml https://mirror.nm.gist.ac.kr/getWordpress
```

Using the template, deploy containers(pods) to run wordpress application

```
kubectl create secret generic mysql-pass --from-literal=password=YOUR PASSWORD  
kubectl apply -f wordpress.yaml
```

Check the pods are running well (About 1 minute later)

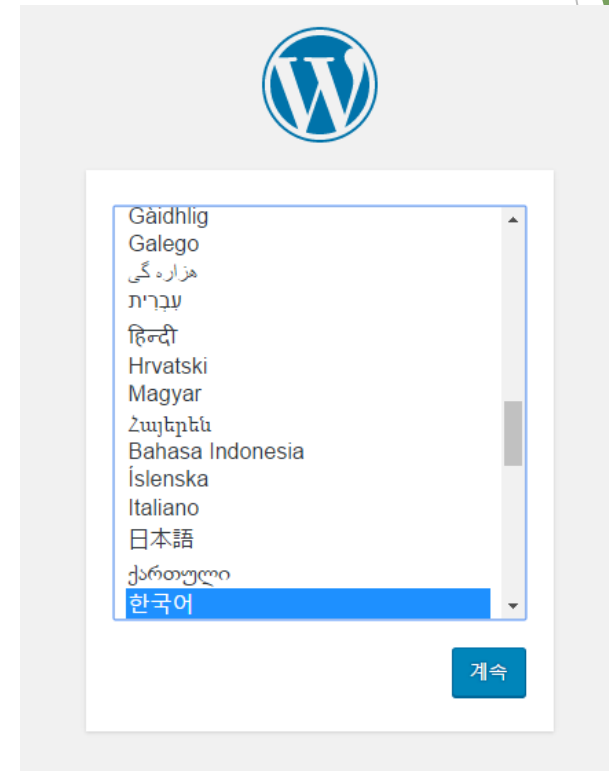
```
kubectl get svc -o wide
```

Running Web Application

- Run Wordpress application (2)



Try
<http://your NUC IP:30080>



Appendix

Control Tower: Docker Private Registry

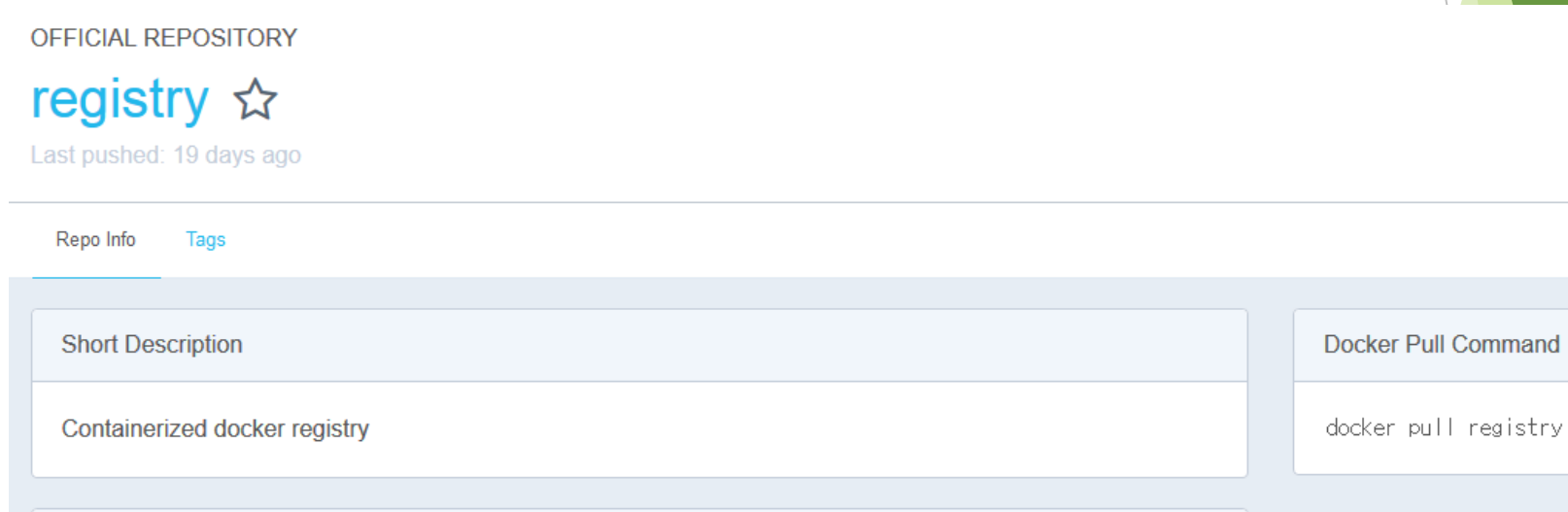
Docker Private Registry

- About

In previous slides, Docker image is operated and maintained like source code (e.g svn, git...)

Also we used Docker public repository, Docker Hub. (which looks like github)

Surely, Docker also provides private repository: Registry.



OFFICIAL REPOSITORY

registry ☆

Last pushed: 19 days ago

Repo Info Tags

Short Description	Docker Pull Command
Containerized docker registry	<code>docker pull registry</code>

Docker Private Registry

- How to use (1)



Deployment(**in control tower**):

```
docker run -d -p 5000:5000 --restart=always --name registry registry:2
```

--restart=always means this container is already running even Docker host rebooted.

Docker Private Registry

- How to use (2)



You may configure insecure-registry options on NUC and Raspberry Pi (Basic registry runs on insecure mode.)

Setting(In NUC and RPi):

```
sudo vi /etc/default/docker
```

```
DOCKER_OPTS="--insecure-registry [control tower ip]:5000"
```

```
# If you need Docker to use an HTTP proxy, it can also be
ere.
#export http_proxy="http://127.0.0.1:3128/"

# This is also a handy place to tweak where Docker's tempo
go.
#export TMPDIR="/mnt/bigdrive/docker-tmp"
DOCKER_OPTS="--insecure-registry :5000"
```

If any DOCKER_OPTS is already been, just add this phrase to end of OPTS.

```
sudo service docker restart
```



Docker Private Registry

- How to push and pull image (1)

In this time, we make a image of nginx container and push to registry

Making container image and push to registry (In NUC):

```
docker commit nginx  
docker images
```

```
tein@vbox-develop:~$ docker commit nginx  
a67e74d9e4ee5dc194d7449f964e0c093643b74a0fbbebee4d805dac34219294  
tein@vbox-develop:~$ docker images  
REPOSITORY          TAG                 IMAGE ID  
    CREATED          VIRTUAL SIZE  
<none>              <none>             a67e74d9e4ee  
    8 seconds ago    248.3 MB  
nginx                latest              9f55a676b5c2
```

Docker Private Registry

- How to push and pull image (2)



Tagging image for pushing and push image

```
docker tag nginx [control tower ip]:5000/nginx  
docker push [control tower ip]:5000/nginx
```

```
tein@vbox-develop:~$ docker tag nginx :5000/nginx  
tein@vbox-develop:~$ docker push :5000/nginx  
The push refers to a repository [ :5000/nginx] (len: 9f55a676b5c2: Pushed  
35a2943903f2: Pushed
```

Check!

```
curl [control tower ip]:5000/v2/_catalog
```

```
tein@vbox-develop:~$ curl :5000/v2/_catalog  
{  
  "repositories": [  
    "arm_flume_vis_agent",  
    "arm_node_example",  
    "kafka",  
    "kafka_origin",  
    "kafka_server",  
    "mktopic",  
    "nginx",  
    "u",  
    "vis2016",  
    "zookeeper"  
  ]  
}
```

You can find!

Docker Private Registry

- How to push and pull image (3)

Pulling(In other machines):

```
docker pull [control tower ip]:5000/nginx
```

```
bjj@orche:~$ docker pull :5000/nginx
Using default tag: latest
latest: Pulling from nginx
efd26ecc9548: Extracting 36.18 MB/51.34 MB
a3ed95caeb02: Download complete
a48df1751a97: Download complete
8ddc2d7beb91: Download complete
```

Thank You for
Your Attention
Any Questions?

