

Jump into DevOps World

22-23 September 2018

Before start

Glithub: [https://github.com/lma8/Jump into DevOps world](https://github.com/lma8/Jump_into_DevOps_world)

digitalocean: [http://bit.ly/do get 100](http://bit.ly/do_get_100) Free \$100

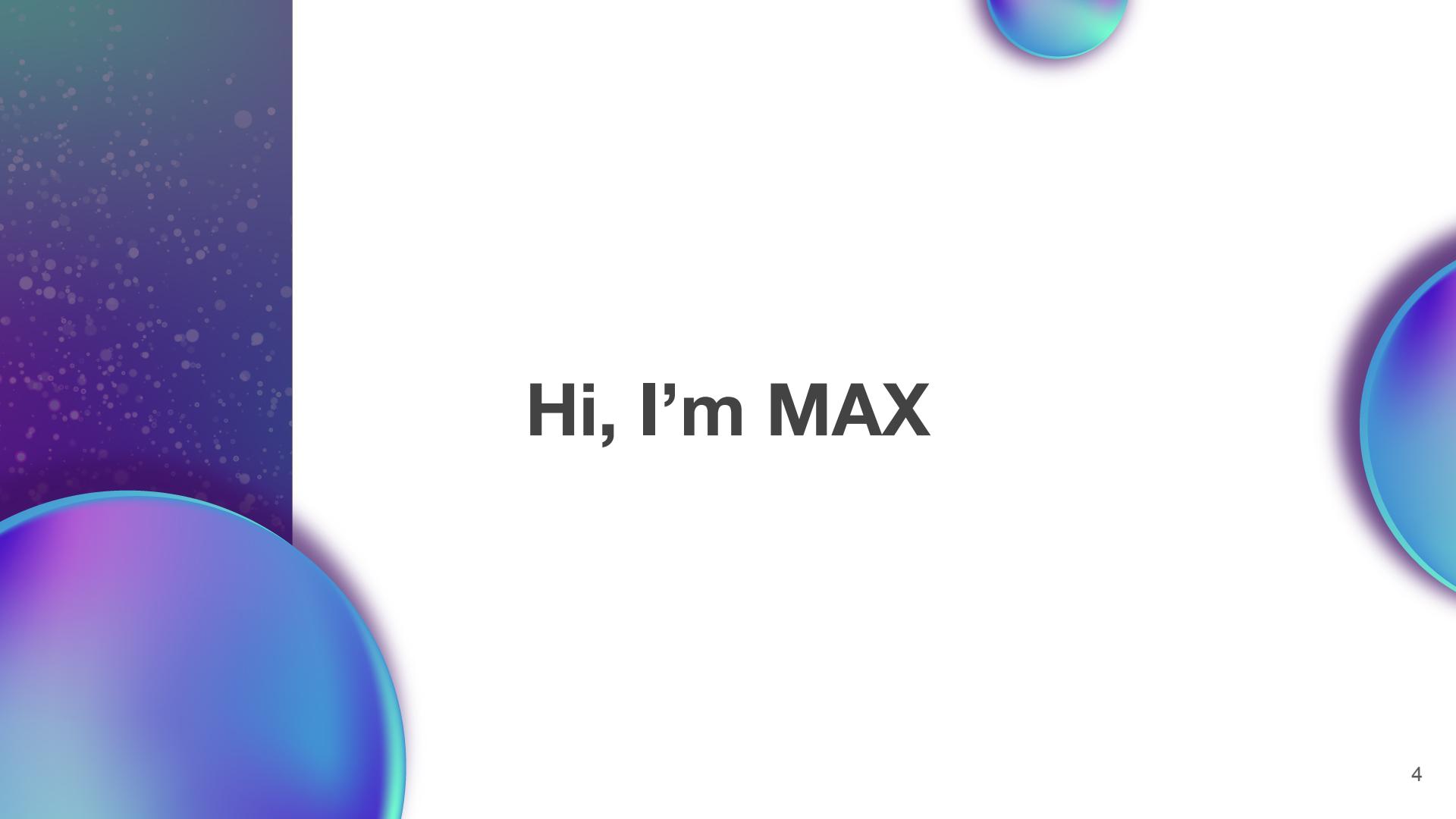
putty: <https://www.ssh.com/ssh/putty/>

Mysql workbench: <https://www.mysql.com/products/workbench/>

Docker CE: <https://www.docker.com/get-started>

node js 8.12: <https://nodejs.org/en/download/>

แมวไม่ว่าจะสีอะไรขอให้จับหนูได้ก็เป็นพ่อ



Hi, I'm MAX

คาดหวังอะไรจากคอร์ส呢 ?

“

Business is Happy

“

Everyone is Happy

“

Programmer is Happy

What is DevOps ?

“

‘DevOps’ ไม่ใช่แค่เรื่องของ ‘Automation’

“

‘DevOps’ ไม่ใช่แค่เรื่องของ ‘Automation’

‘DevOps’ ไม่ใช่เรื่องของการใช้ ‘Tools’

“

“DevOps” = “culture”

the DevOps methodology is not about technical capability.
Successful DevOps is an organizational challenge.

“

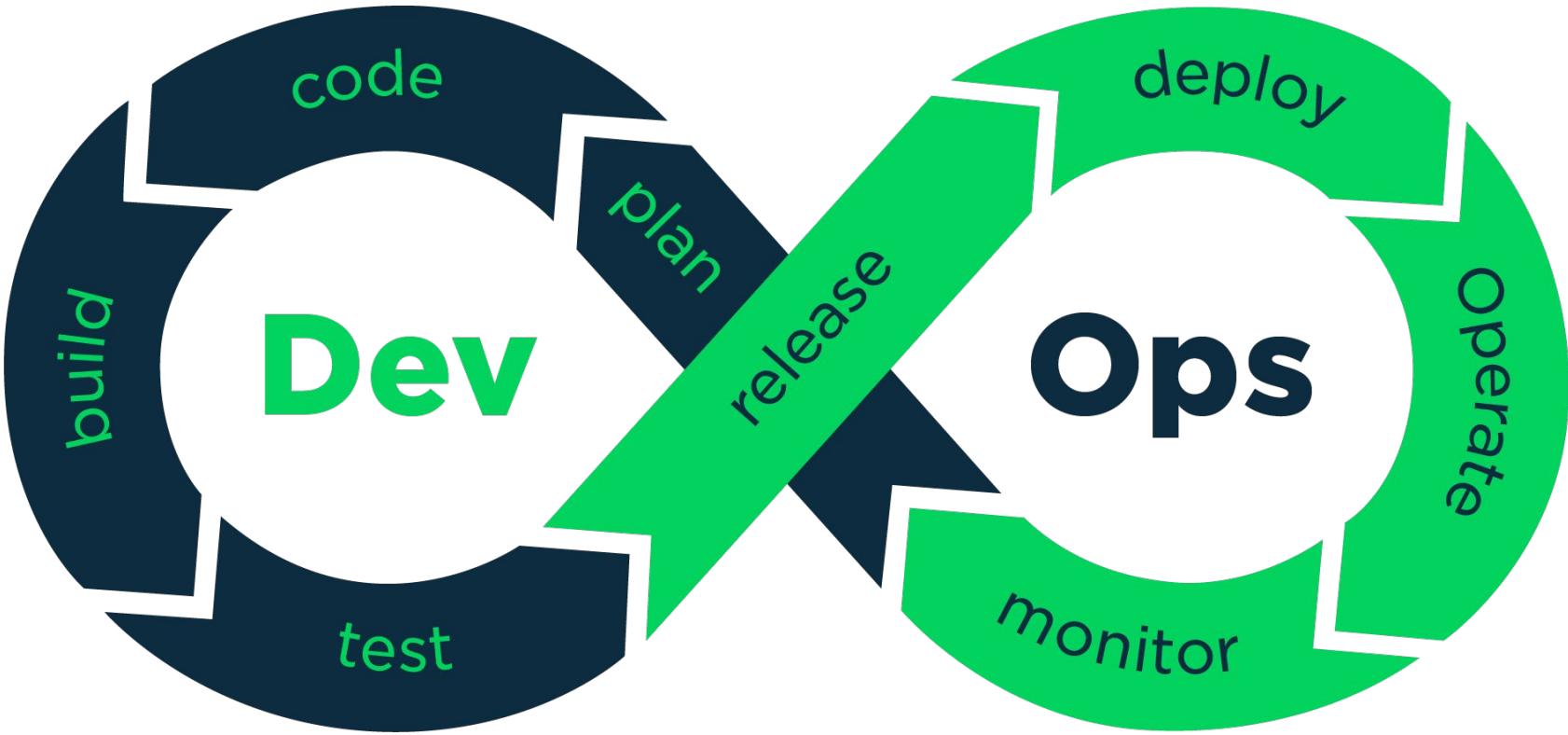
DevOps is a culture, not a role!
The whole company needs
to be doing DevOps for it to work.

Mike Dilworth

“

Same Goal

Deliver software faster and more reliably to customers

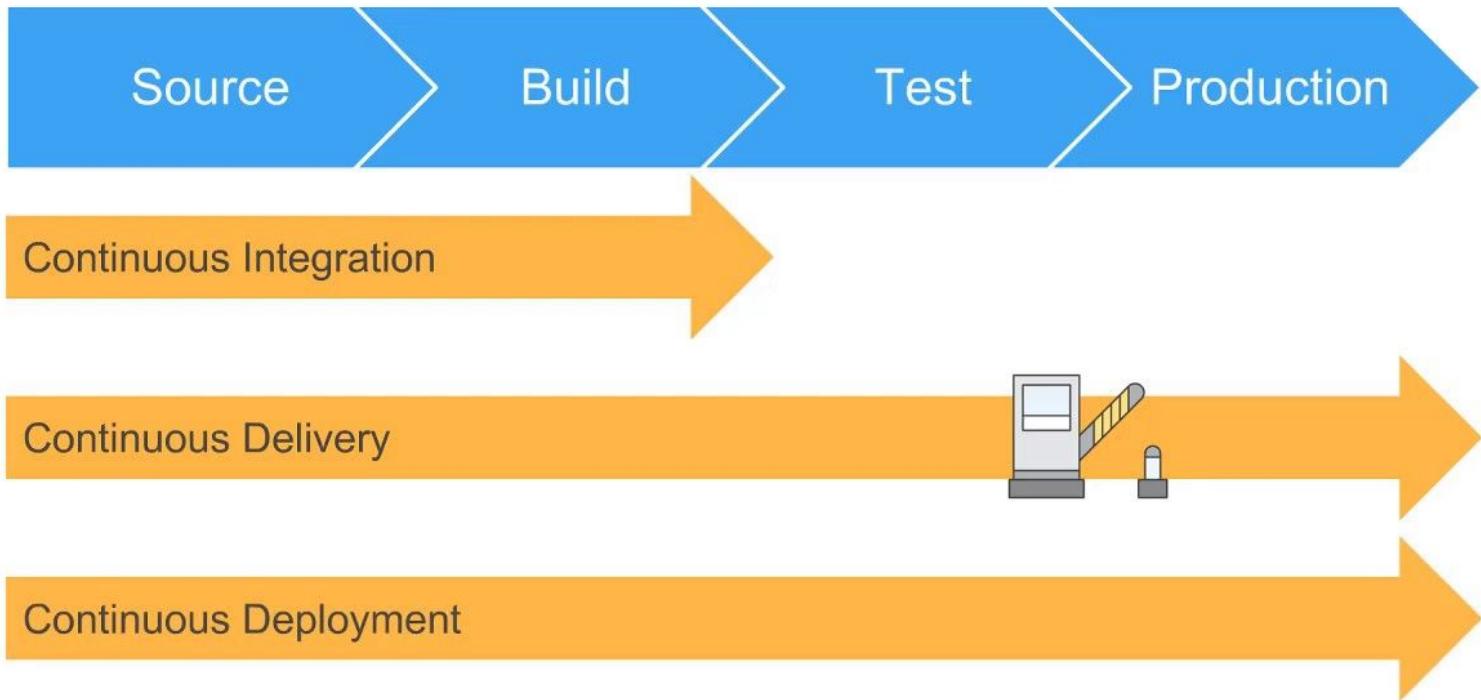


What you should focus?

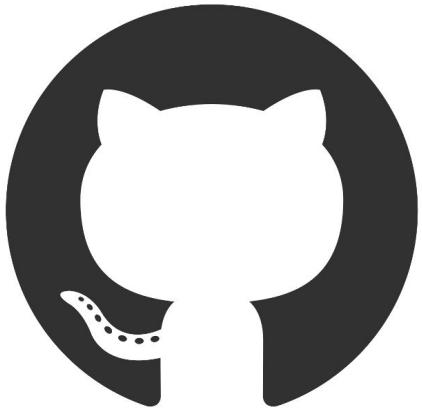
- High availability
- Respect production
- งานน่าเบื่อ ต้องไม่ทำเอง มีเวลา空余ไปฟอกสี Coding
- อะไรจะทำให้ Dev ปวดหัว ต้องไม่เกิดขึ้น!

Schedule

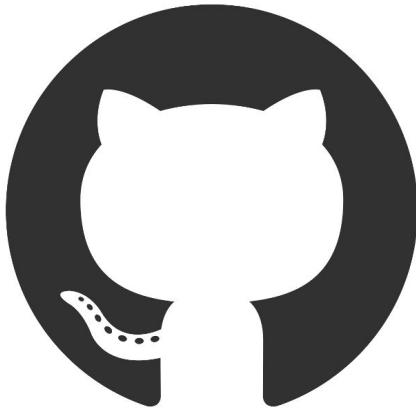
- What DevOps is
- What is CI/CD
- Docker
- Docker-compose
- Setup your MySql cluster for high availbailty (MariaDB Galera Cluster)
- Rancher
- CircleCI
- ELK for Logging
- API Gateway with Kong

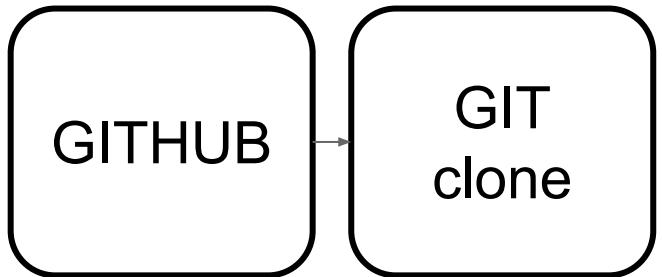


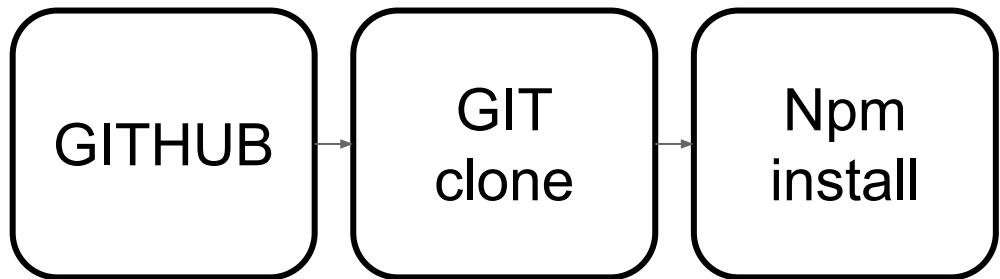
Source
code

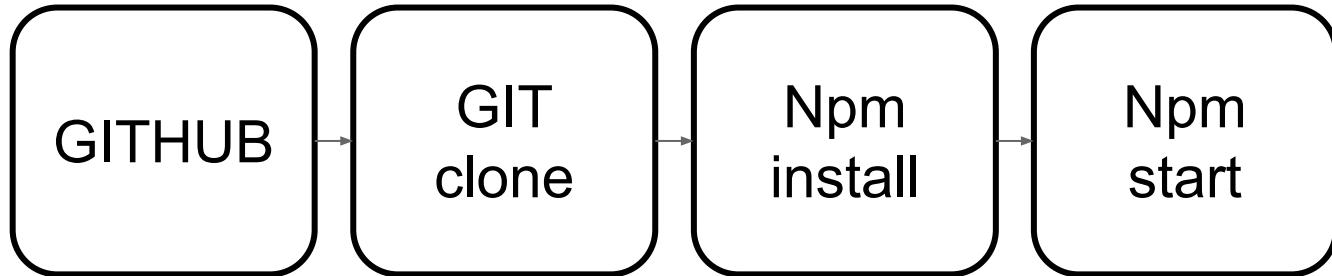


Source
code

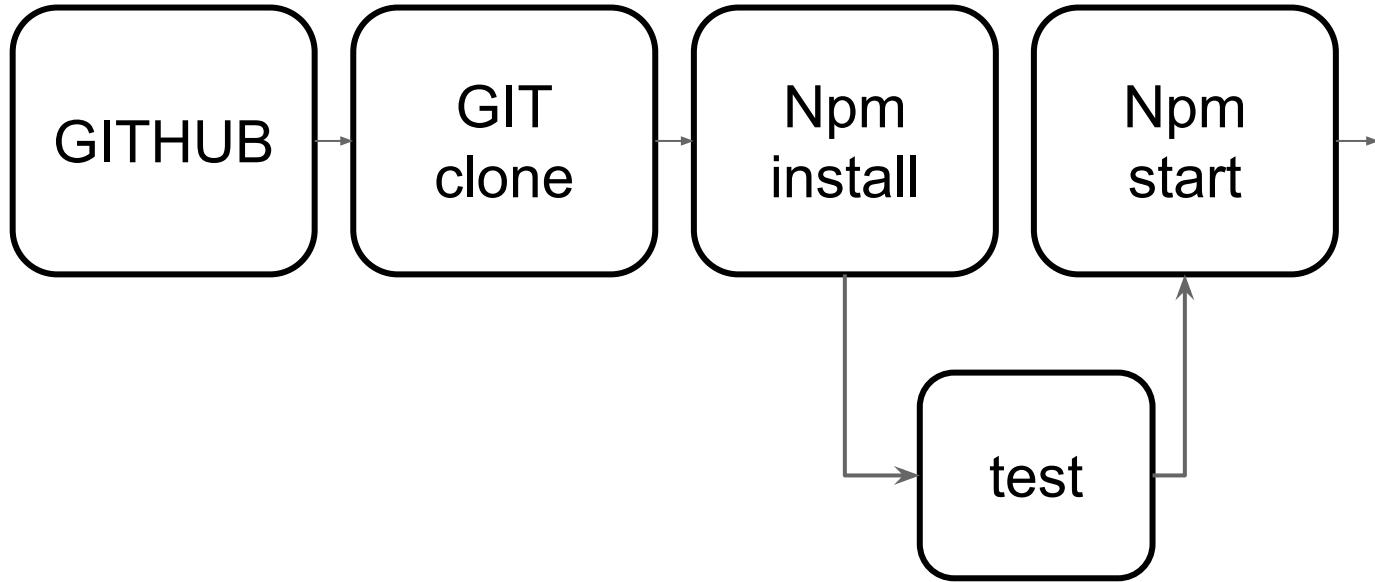


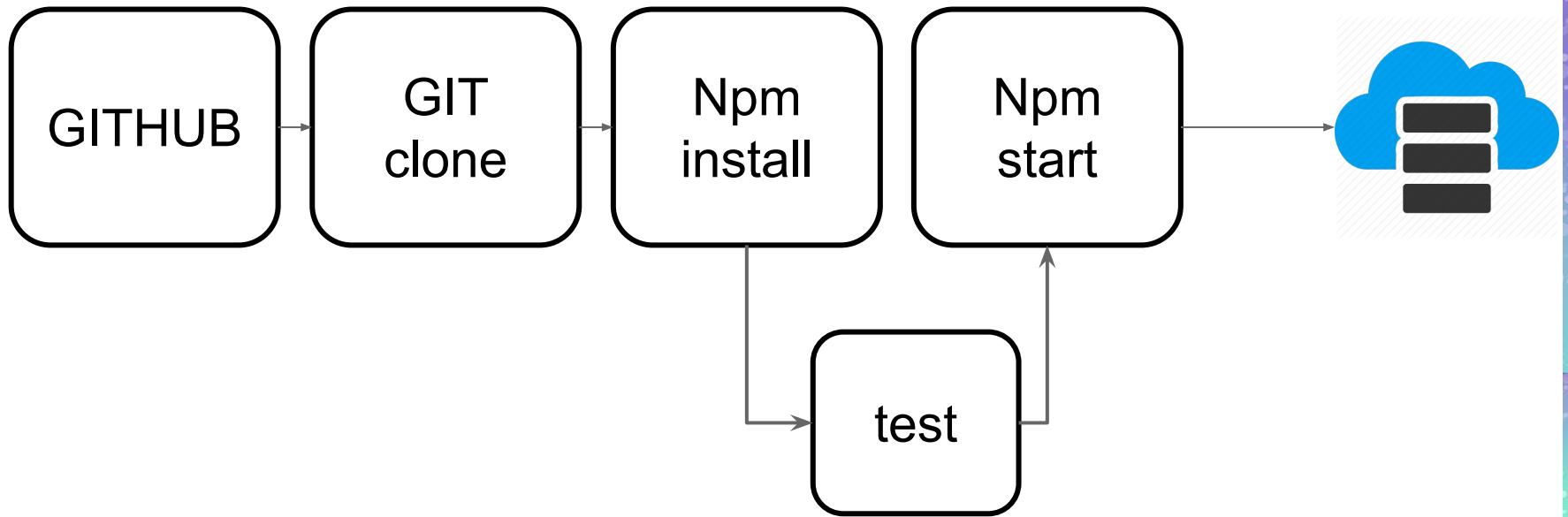


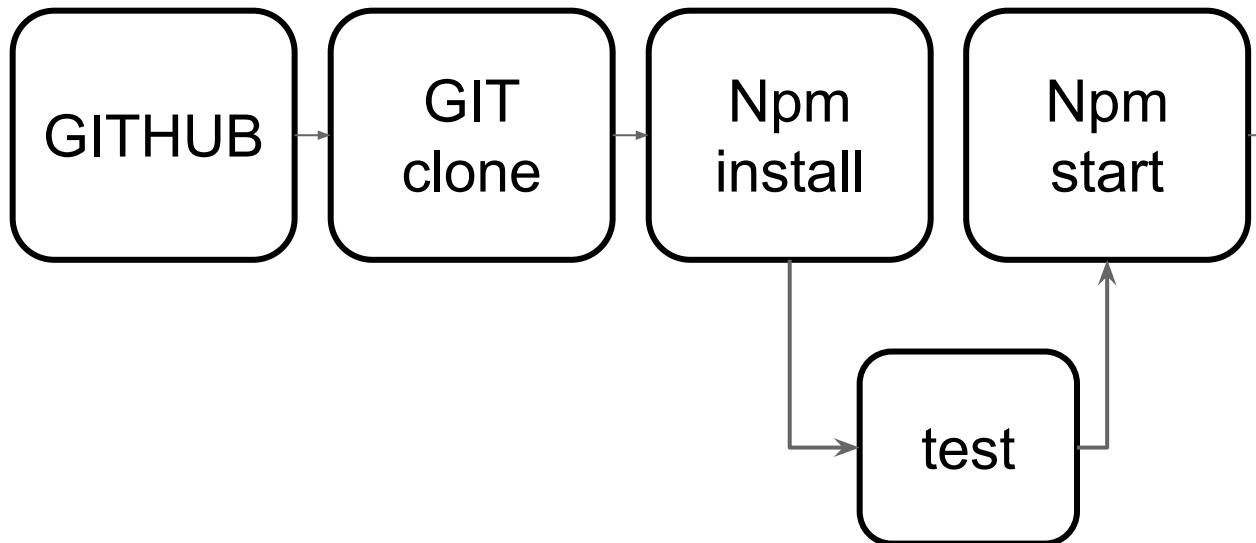


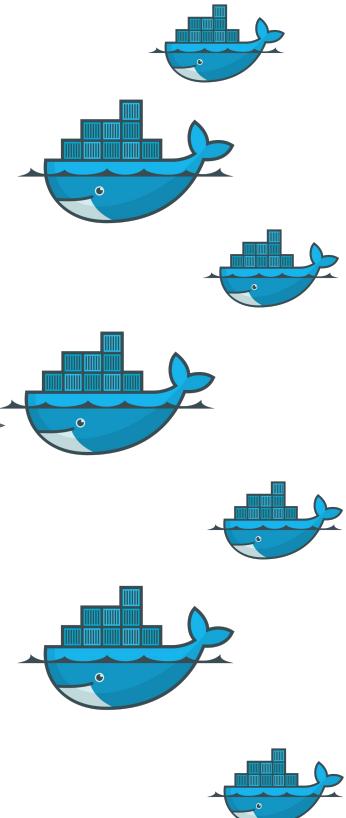
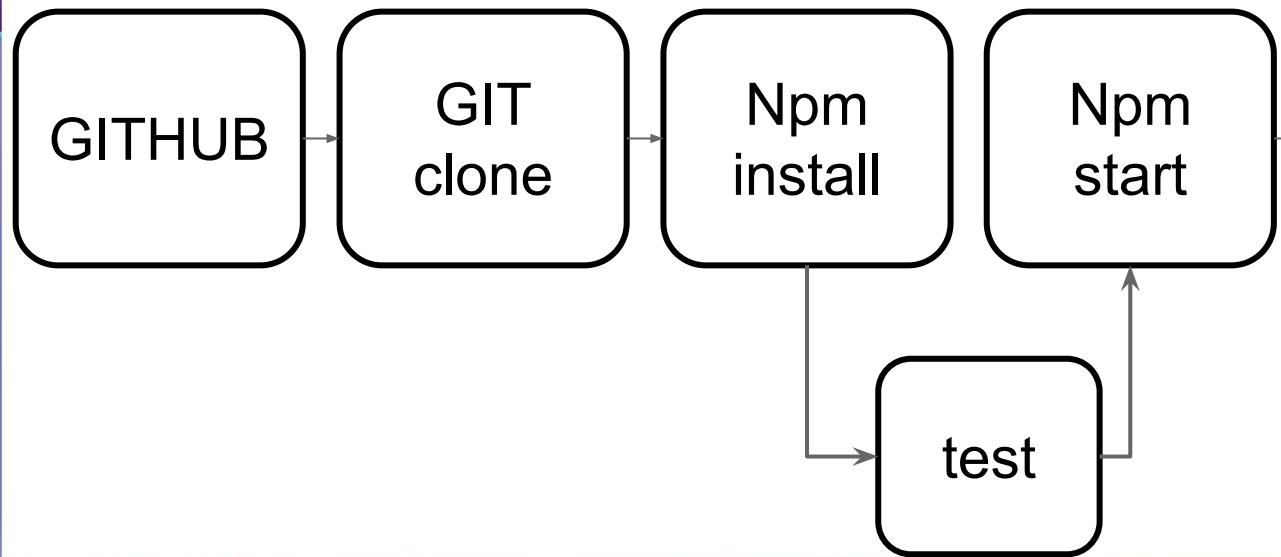


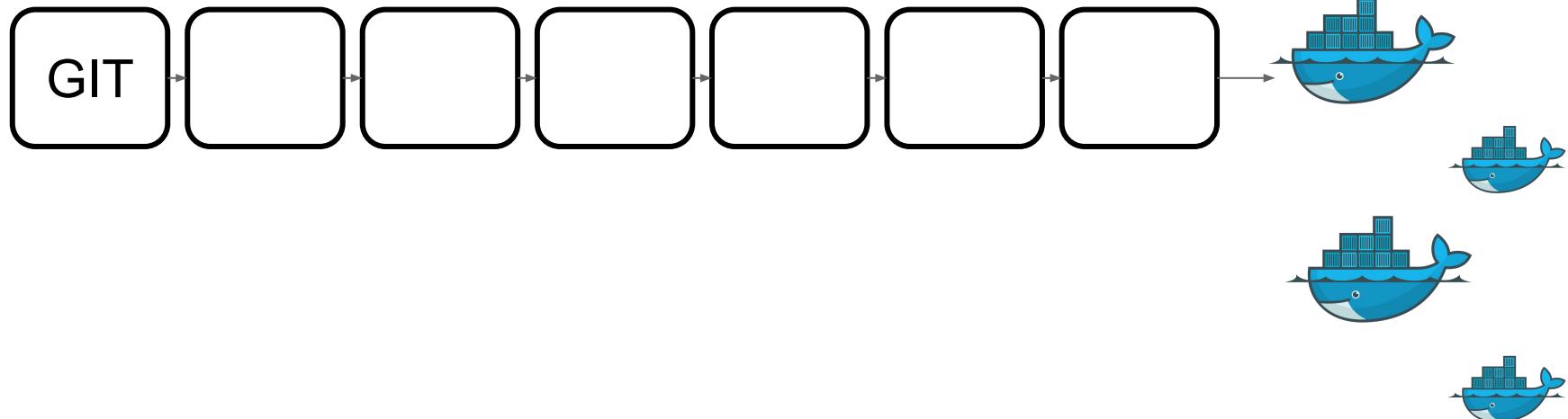


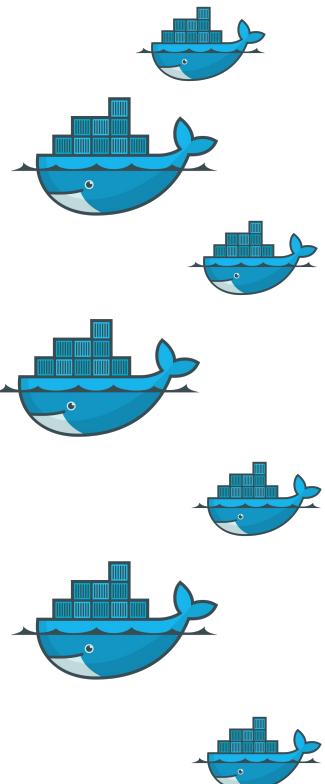
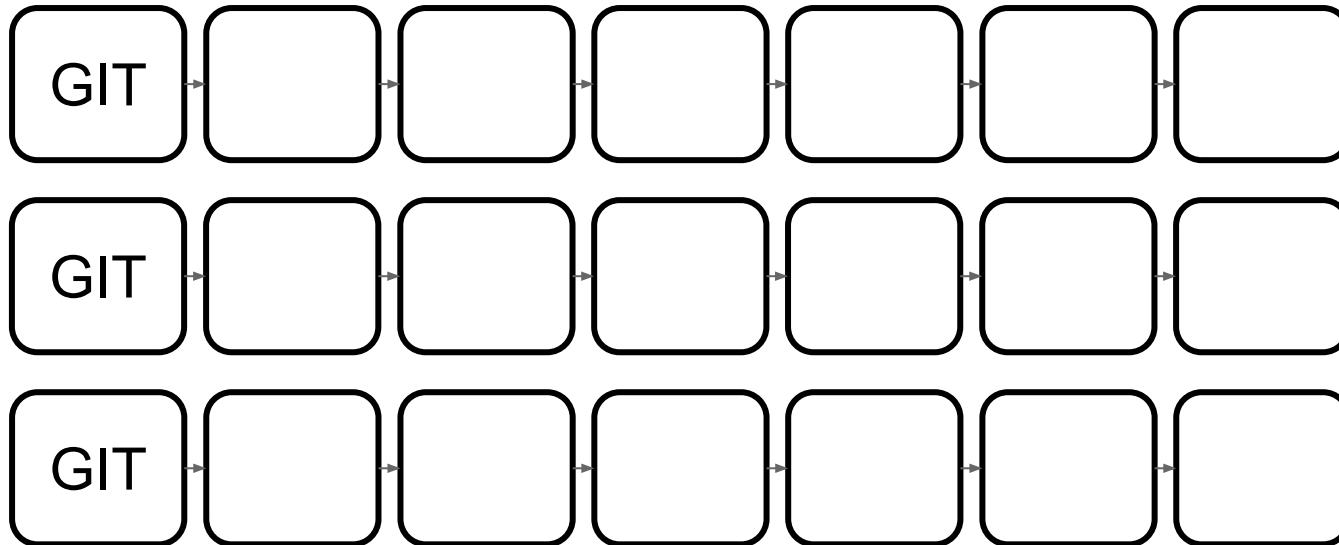












Schedule

- What DevOps is
- What is CI/CD
- Docker
- Docker-compose
- Setup your MySql cluster for high availbailty (MariaDB Galera Cluster)
- Rancher
- CircleCI
- ELK for Logging



continuous integration and delivery platform

- Configuration
 - .yml
- Fast Builds



RANCHER

- Infrastructure orchestration
- Container orchestration and scheduling
- Application catalog
- enterprise-grade control

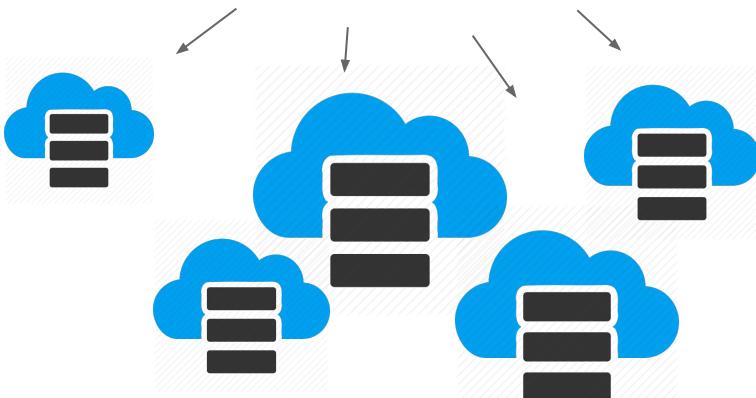


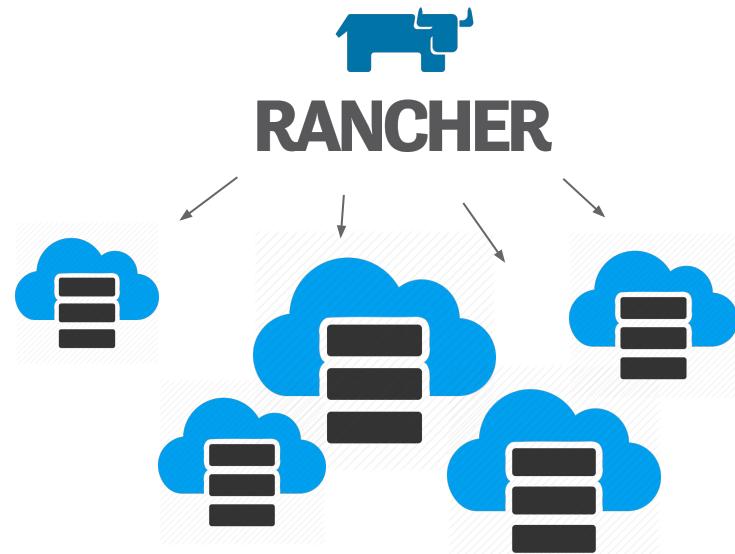
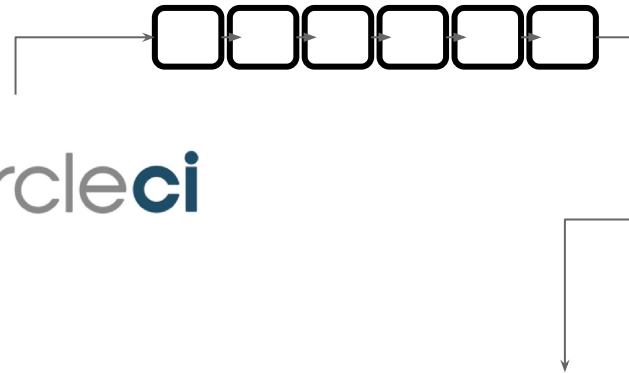
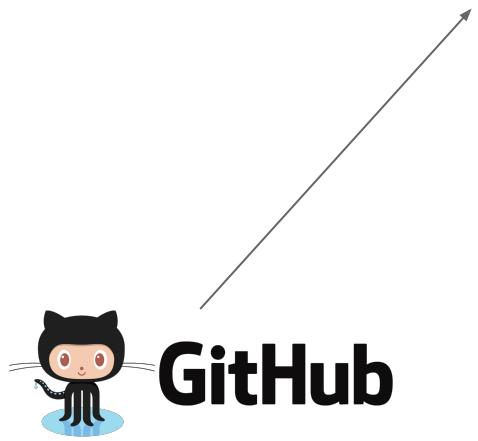
RANCHER

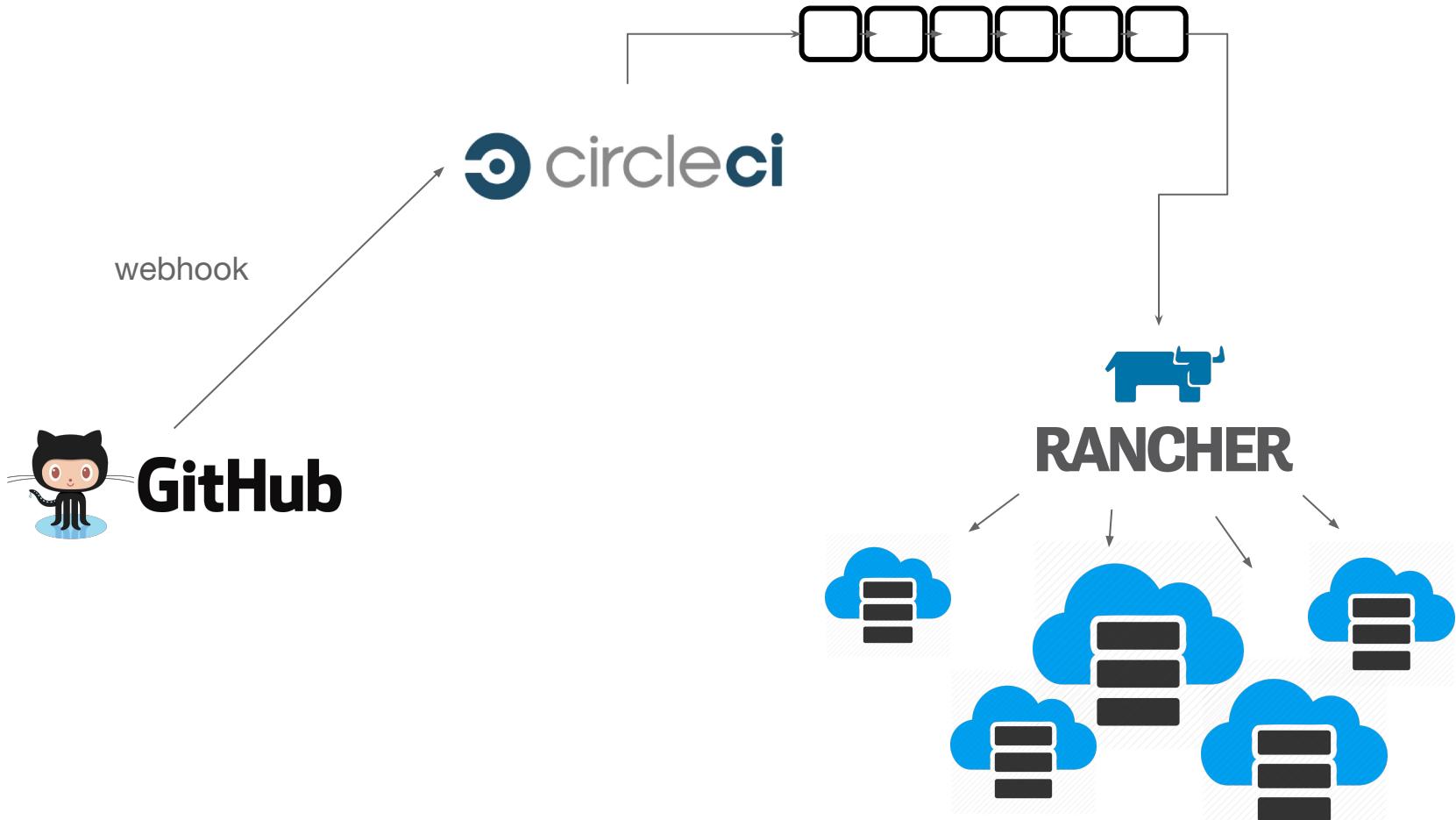
1. Rancher 1.6
 - a. **Cattle** , Docker swarm, Kubernetes
2. Rancher 2.0
 - a. Your Enterprise Kubernetes Platform

Hosts [Add Host](#) Show System [...](#)

ACTIVE After5-Data-Pipeline-1 209.97.1 1.12.6 Ubuntu 16.04.4 LTS (with KVM) (4.4.0) 2x2.2 GHz 3.86 GiB 77.4 GiB ↳ digitalocean message_broker=true redis=true	ACTIVE After5-Data-Pipeline-2 209.97.1 1.12.6 Ubuntu 16.04.4 LTS (with KVM) (4.4.0) 2x2.2 GHz 3.86 GiB 77.4 GiB ↳ digitalocean message_broker=true redis=true	ACTIVE After5-database-1 139.5 1.12.6 Ubuntu 16.04.3 LTS (with KVM) (4.4.0) 2.2 GHz 3.86 GiB 48.3 GiB ↳ digitalocean database=true worker=true	ACTIVE After5-database-2 159.65.1 1.12.6 Ubuntu 16.04.3 LTS (with KVM) (4.4.0) 2.2 GHz 3.86 GiB 48.3 GiB ↳ digitalocean database=true worker=true	ACTIVE After5-LB1 139.59 17.06.2-ce Ubuntu 16.04.4 LTS (with KVM) (4.4.0) 2x2.4 GHz 1.95 GiB 58 GiB ↳ digitalocean database=true LB=true	ACTIVE After5-LB2 128.199 17.06.2-ce Ubuntu 16.04.4 LTS (with KVM) (4.4.0) 2x2.4 GHz 1.95 GiB 58 GiB ↳ digitalocean database=true LB=true	ACTIVE After5-staging 206.189 1.12.6 Ubuntu 16.04.4 LTS (with KVM) (4.4.0) 2x2.2 GHz 3.86 GiB 58 GiB ↳ digitalocean staging=true
Stack: healthcheck healthcheck-13 10.42.47.6	Stack: healthcheck healthcheck-11 10.42.97.233	Stack: backend-api Main-API-3 10.42.105.172	Stack: backend-api Main-API-4 10.42.206.188	Stack: database galera-3 10.42.213.135	Stack: database galera-4 10.42.50.205	Stack: healthcheck healthcheck-10 10.42.108.154
Stack: ipsec cni-driver-11 None ipsec-13 10.42.136.98 Sidekicks 2/2	Stack: ipsec cni-driver-9 None ipsec-11 10.42.137.176 Sidekicks 2/2	Stack: database galera-2 10.42.15.224 Sidekicks 3/3	Stack: database galera-1 10.42.2.46 Sidekicks 3/3	Stack: Default hello-world-1 10.42.218.58 LoadBalancer-1 10.42.232.21	Stack: database galera-4 10.42.94.181 galera-lb-2 10.42.87.148 galera-lb-1 10.42.219.89	Stack: ipsec cni-driver-8 None ipsec-10 10.42.186.0 Sidekicks 2/2
Stack: network-services metadata-15 172.17.0.2 Sidekicks 2/2 network-manager-14 None	Stack: network-services metadata-13 172.17.0.2 Sidekicks 2/2 network-manager-12 None	Stack: healthcheck healthcheck-7 10.42.249.119	Stack: gateway-external api-gateway-ext... 10.42.107.27	Stack: healthcheck healthcheck-8 10.42.147.238	Stack: ipsec cni-driver-2 None ipsec-5 10.42.140.243 Sidekicks 2/2	Stack: network-services network-manager-11 None metadata-12 172.17.0.2 Sidekicks 2/2
Stack: rabbitmq-3 rabbitmq-2 10.42.120.68 Sidekicks 2/2	Stack: rabbitmq-3 rabbitmq-1 10.42.26.5 Sidekicks 2/2	Stack: ipsec cni-driver-3 None ipsec-5 10.42.140.243 Sidekicks 2/2	Stack: netdata netdata-3 10.42.41.82 Sidekicks 1/2	Stack: ipsec cni-driver-4 None ipsec-6 10.42.21.245 Sidekicks 2/2	Stack: ipsec cni-driver-9 10.42.117.14 Sidekicks 2/2 cni-driver-7 None	Stack: ipsec cni-driver-1 None ipsec-3 10.42.167.123 Sidekicks 2/2
Stack: redis	Stack: redis	Stack: netdata netdata-3 10.42.41.82 Sidekicks 1/2	Stack: netdata netdata-3 10.42.41.82 Sidekicks 1/2	Stack: network-services metadata-11 172.17.0.2 Sidekicks 2/2	Stack: scheduler scheduler-1 10.42.184.205 Sidekicks 2/2	Stack: Staging galera-1 10.42.44.100 Sidekicks 2/2



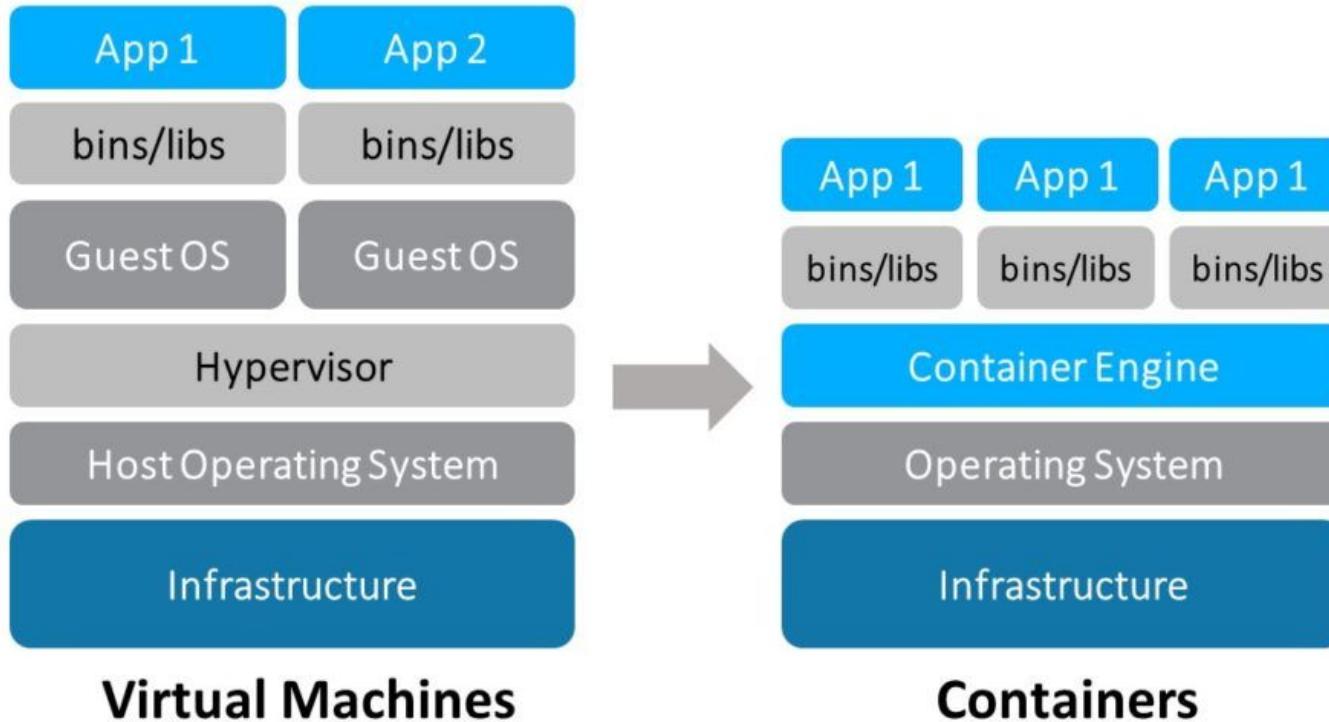




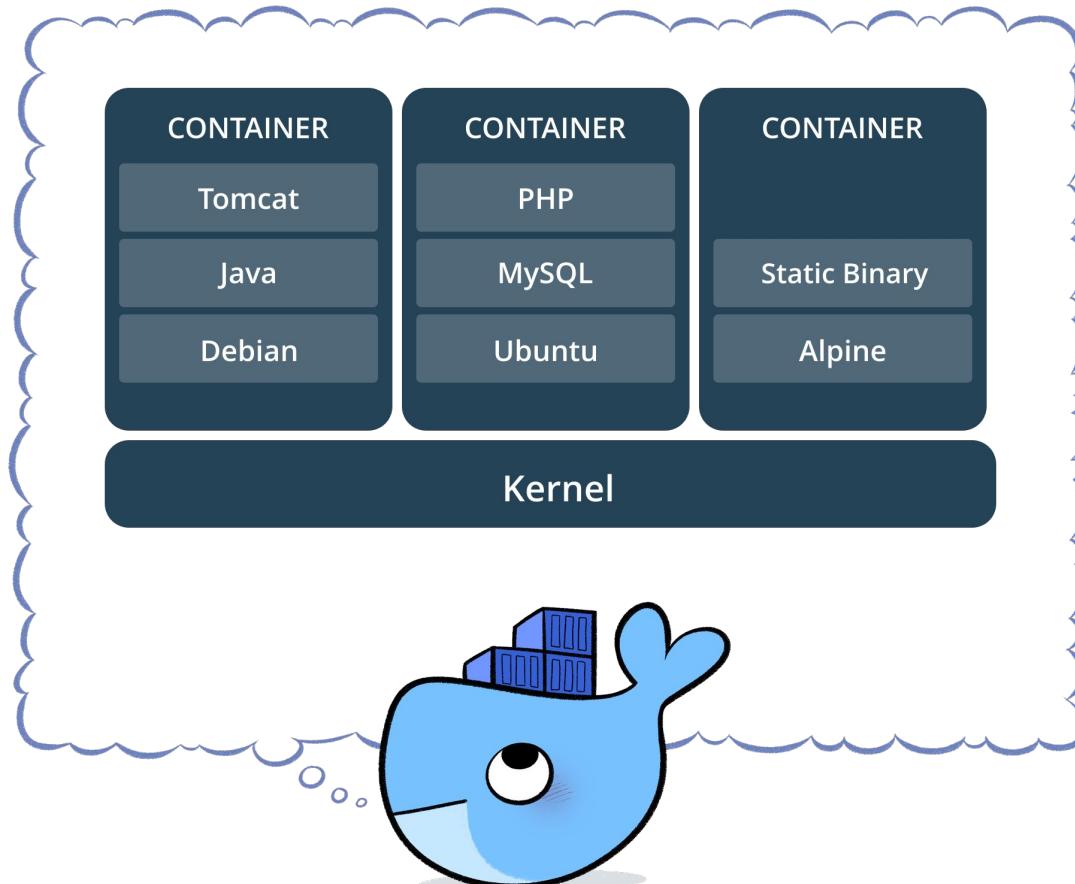
4 Steps for make your pipeline

1. Prepare your Rancher server
2. Add new server (worker)
3. Dockerfile
4. Circleci config

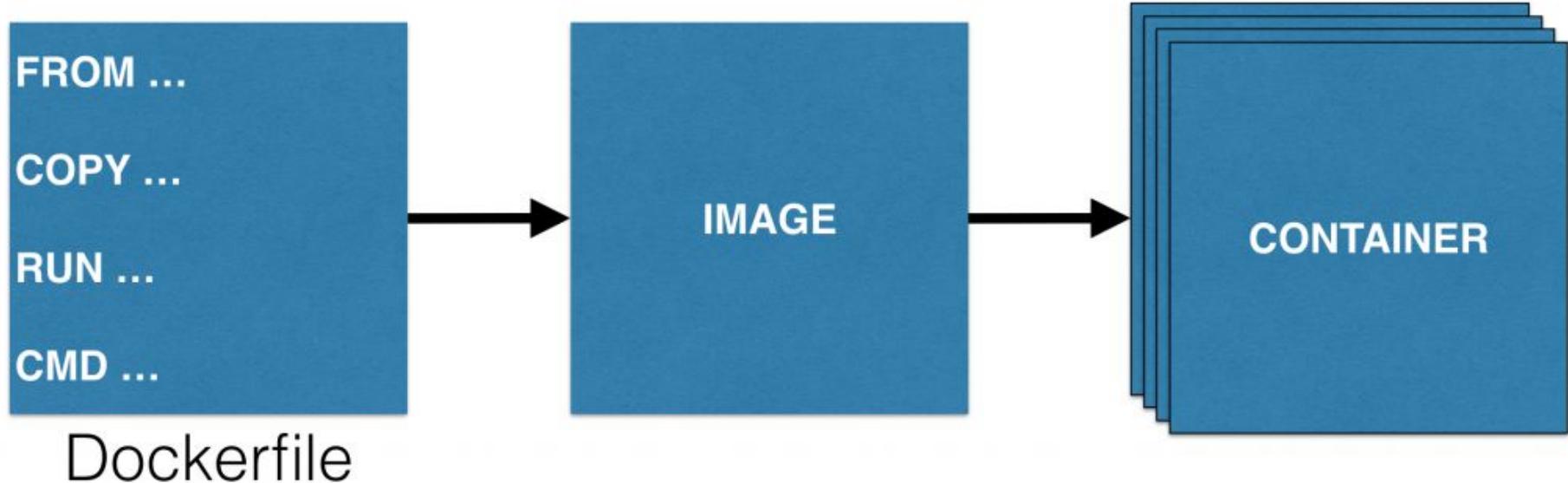
Docker vs VM



Docker



Docker lifecycle



Docker Basic command

- docker run {image}
 - i.e. docker run ima8/demo1
- docker run -d {image}
- docker run -p 8080:80 {image}
- docker run --name demo {image}
- docker run --name demo -p 8080:80 {image}

Demo: Docker

```
“ docker run -p 8080:80 nginx ”
```

Docker: volume

- -v /app:/home/app
 - -v /app:./
- docker run -v /app:/home/app -d ima8/demo1
- docker logs {id}
- docker exec -it {id} bash

Cheat Sheet



RUN

Initialize swarm mode and listen on a specific interface
`docker swarm init --advertise-addr 10.1.0.2`

Join an existing swarm as a manager node
`docker swarm join --token <manager-token> 10.1.0.2:2377`

Join an existing swarm as a worker node
`docker swarm join --token <worker-token> 10.1.0.2:2377`

List the nodes participating in a swarm
`docker node ls`

Create a service from an image exposed on a specific port and deploy 3 instances
`docker service create --replicas 3 -p 80:80 --name web nginx`

List the services running in a swarm
`docker service ls`

Scale a service
`docker service scale web=5`

List the tasks of a service
`docker service tasks web`

`docker run`
 `--rm` remove container automatically after it exits
 `-it` connect the container to terminal
 `--name web` name the container
 `-p 5000:80` expose port 5000 externally and map to port 80
 `-v ~/dev:/code` create a host mapped volume inside the container
 `alpine:3.4` the image from which the container is instantiated
 `/bin/sh` the command to run inside the container

Stop a running container through SIGTERM
`docker stop web`

Stop a running container through SIGKILL
`docker kill web`

Create an overlay network and specify a subnet
`docker network create --subnet 10.1.0.0/24 --gateway 10.1.0.1 -d overlay mynet`

List the networks
`docker network ls`

List the running containers
`docker ps`

Delete all running and stopped containers
`docker rm -f $(docker ps -aq)`

Create a new bash process inside the container and connect it to the terminal
`docker exec -it web bash`

Print the last 100 lines of a container's logs
`docker logs --tail 100 web`

BUILD

SHIP

Build an image from the Dockerfile in the current directory and tag the image
`docker build -t myapp:1.0 .`

List all images that are locally stored with the Docker engine
`docker images`

Delete an image from the local image store
`docker rmi alpine:3.4`

Pull an image from a registry
`docker pull alpine:3.4`

Retag a local image with a new image name and tag
`docker tag alpine:3.4 myrepo/myalpine:3.4`

Log in to a registry (the Docker Hub by default)
`docker login my.registry.com:8000`

Push an image to a registry
`docker push myrepo/myalpine:3.4`

Workshop: Docker I

“ run nginx website with docker and edit the content ”

nginx path: /usr/share/nginx/html

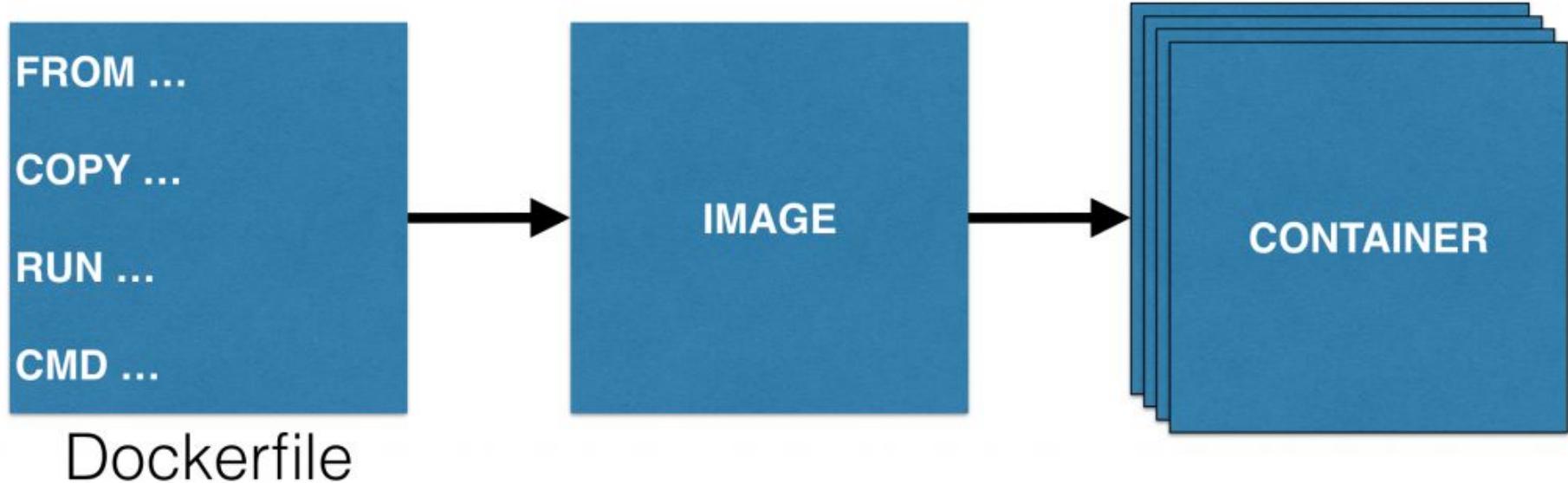
Workshop: Docker I

“ run nginx website with docker and edit the content ”

nginx path: /usr/share/nginx/html

```
docker run -v ~/nginx:/usr/share/nginx/html -p 80:80 -d nginx
```

Docker lifecycle



Docker Basic command

- docker build .
 - docker build -t ima8/demo1 .
 - docker build -f Dockerfile.staging .
 - docker build -t {username}/{project_name} -f Dockerfile.staging .
- docker run -p 80:80 ima8/demo1 .
- docker images

Workshop: Dockerfile I

“ Make your own Dockerfile ”

```
1  FROM nginx
2
3  COPY index.html /usr/share/nginx/html
4
5  EXPOSE 80
6
7
8  |
```

Workshop: Dockerfile II

“ Make your docker file and CMD ”

<https://github.com/lma8/example-nodejs-express>

Docker File

```
1 FROM node:8.11-alpine          # Base image
2
3
4 RUN mkdir -p /app
5
6
7 COPY / /app
8
9
10 WORKDIR /app
11
12
13 CMD [ "node", "/app/server.js" ]    # run command
14
15
16 EXPOSE 3000                  # Post
```

Guideline

- docker build -t ima8/demo2 .
- docker run -p 8000:3000
ima8/demo2

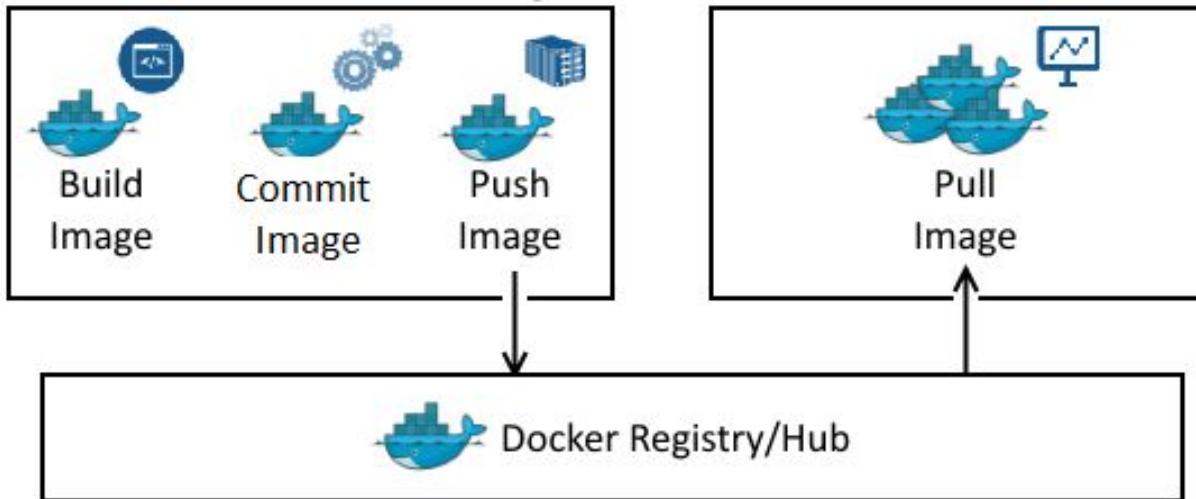
```
Dockerfile x
1  FROM node:8.11-alpine
2
3  RUN mkdir -p /app
4
5  COPY thisisapp/ /app
6
7  WORKDIR /app
8
9  RUN npm install
10
11 CMD ["node","/app/bin/www"]
12
13 EXPOSE 3000
14
```

Workshop: Dockerfile III

“ Make your docker file and run it with volume ”

github.com/lma8/node-express-multer-image-upload

Docker Registry



Docker Registry

- Register at <https://hub.docker.com/>
- docker login
- docker build -t {username}/{name} .
 - i.e. docker build -t ima8/demo .
 - docker tag {image_name} {username}/{name}
- docker push {username}/{name}

Docker Registry

- docker pull {username}/{name}
- docker run {username}/{name}

Workshop: Docker Registry I

“ ให้เอา Workshop docker file ก่อนหน้า
ขึ้น Docker Registry ”

Docker Private Registry

- Gitlab.com
- Self Hosted (gitlab omnibus)



Gitlab Docker Private Registry

```
# docker login my.gitlab.com  
# docler build -t my.gitlab.com/api/foo .  
# docker pull my.gitlab.com/api/foo
```

Gitlab Registry

```
1 version: '3.6'
2 services:
3   gitlab:
4     image: 'gitlab/gitlab-ce:rc'
5     restart: always
6     hostname: 'gitlab'
7     environment:
8       GITLAB_OMNIBUS_CONFIG: |
9         external_url 'https://gitlab.local'
10        nginx['ssl_certificate'] = "/etc/gitlab/ssl/gitlab.crt"
11        nginx['ssl_certificate_key'] = "/etc/gitlab/ssl/gitlab.key"
12        registry_external_url 'https://registry.local'
13        gitlab_rails['registry_host'] = "registry"
14     ports:
15       - '80:80'
16       - '443:443'
17       - '22:22'
18     volumes:
19       - './config:/etc/gitlab'
20       - './ssl:/etc/gitlab/ssl'
21       - './logs:/var/log/gitlab'
22       - './data:/var/opt/gitlab'
23
24   registry:
25     image: registry:2
26     restart: always
27     hostname: registry
28     volumes:
29       - ./registry:/var/lib/registry
30
```

Self Signed Certificate for Gitlab

```
# mkdir -p ssl  
  
# openssl genrsa -out ssl/gitlab.key 2048  
  
# openssl req -new -key ssl/gitlab.key -out ssl/gitlab.csr  
  
# openssl x509 -req -days 365 \  
    -in ssl/gitlab.csr \  
    -signkey ssl/gitlab.key \  
    -out ssl/gitlab.crt
```

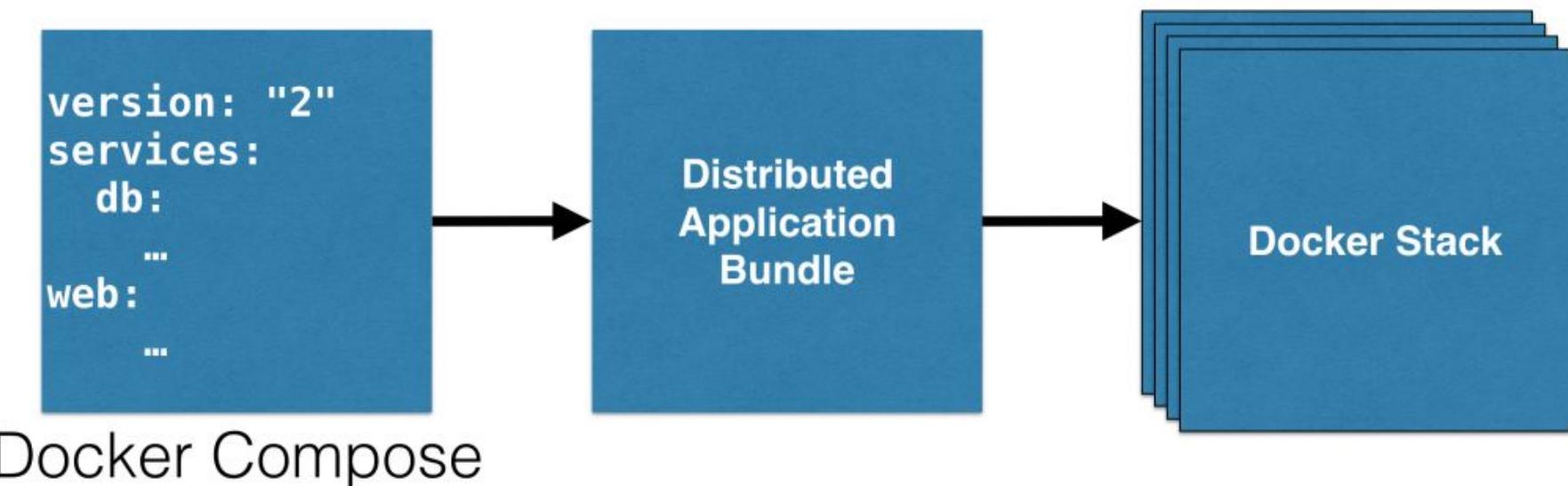
Self Signed Certificate for Registry

```
# openssl genrsa -out ssl/registry.key 2048  
  
# openssl req -new -key ssl/registry.key \  
    -out ssl/registry.csr  
  
# openssl x509 -req -days 365 \  
    -in ssl/registry.csr \  
    -signkey ssl/registry.key \  
    -out ssl/registry.crt
```

Schedule

- What DevOps is
- What is CI/CD
- Docker
- Docker-compose
- Setup your MySql cluster for high availbailtiy (MariaDB Galera Cluster)
- Rancher
- CircleCI
- ELK for Logging

Docker compose



Docker Registry

https://hub.docker.com/_/wordpress/

Docker compose

```
1  version: '2'  
2  
3  services:  
4    wp:  
5      image: wordpress  
6      environment:  
7        - WORDPRESS_DB_HOST=db  
8        - WORDPRESS_DB_PASSWORD=mypass  
9      volumes:  
10        - /home/user/workplace/app:/var/www/html  
11      ports:  
12        - 80:80  
13
```

Docker compose

```
version: '2'

services:
  wp:
    image: wordpress
    environment:
      - WORDPRESS_DB_HOST=db
      - WORDPRESS_DB_PASSWORD=mypass
    volumes:
      - ./app:/var/www/html
    ports:
      - 80:80
```

Docker compose

```
1 version: '2'  
2  
3 services:  
4   wp:  
5     image: wordpress  
6     environment:  
7       - WORDPRESS_DB_HOST=db  
8       - WORDPRESS_DB_PASSWORD=mypass  
9     volumes:  
10      - /home/user/workplace/app:/var/www/html  
11   ports:  
12     - 80:80  
13
```

```
docker-compose start  
docker-compose stop
```

```
docker-compose pause  
docker-compose unpause
```

```
docker-compose ps  
docker-compose up  
docker-compose down
```

Docker compose

docker-compose up vs docker-compose start

- docker-compose start = Starts existing
- docker-compose up = build , create , start

Docker compose

- docker-compose up
 - docker-compose up -d
 - docker-compose -f {docker-compose.yml} up
- docker-compose down
- docker-compose logs
 - docker-compose logs -f
 - follow the logs

Docker compose

- `docker-compose up --force-recreate`
- `docker-compose down && docker-compose build --no-cache && docker-compose up`

Workshop: Docker-compose I

“ run example wordpress project “

path: /ch_docker-compose/1_wordpress

wordpress docker-compose

Workshop: Docker-compose II

“ run your own dockerfile with docker-compose “

```
version: '2'

services:
  webserver:
    build: .
    ports:
      - 8080:80
```

Workshop: Docker-compose III

“ run your own image with docker-compose ”

4 Steps for make your pipeline

1. Prepare your Rancher server
2. Add new server (worker)
3. Dockerfile
4. Circleci config

Prepare your Rancher ecosystem

- 1. Rancher node**
- 2. Database node (mysql)**

Prepare your Rancher ecosystem

Rancher node

1. SINGLE CONTAINER
2. SINGLE CONTAINER - EXTERNAL DATABASE
3. SINGLE CONTAINER - BIND MOUNT MYSQL VOLUME
4. Full Active/Active HA

Prepare your Rancher ecosystem

https://github.com/lma8/Jump_into_DevOps_world

`Jump_into_DevOps_world/ch_setup_rancher/`

- `docker-compose up -d`
- `docker-compose logs -f`

Prepare your Rancher ecosystem

```
mysql:  
  image: mysql:5.7  
  restart: always  
  volumes:  
    - ./var/lib/mysql:/var/lib/mysql  
  ports:  
    - "3306:3306"  
  environment:  
    - MYSQL_ROOT_PASSWORD=your_root_password!  
    - MYSQL_DATABASE=rancher  
    - MYSQL_USER=rancher  
    - MYSQL_PASSWORD=your_password!  
  healthcheck:  
    test: "/usr/bin/mysql --user=rancher --password=your_password! --execute \"SHOW DATABASES;\""  
    interval: 5s  
    timeout: 60s  
    retries: 10
```

Prepare your Rancher ecosystem

```
version: "2.1"
services:
  rancher:
    image: rancher/server:stable
    restart: always
    ports:
      - "8080:8080"
    links:
      - mysql
    depends_on:
      mysql:
        condition: service_healthy
    environment:
      - CATTLE_DB_CATTLE_MYSQL_HOST=mysql ## Your public ip
      - CATTLE_DB_CATTLE_MYSQL_PORT=3306
      - CATTLE_DB_CATTLE_MYSQL_NAME=rancher
      - CATTLE_DB_CATTLE_USERNAME=rancher
      - CATTLE_DB_CATTLE_PASSWORD=your_password!
  mysql:
```

Workshop: rancher I

“ setup rancher on cloud server “

The screenshot shows the Rancher web interface. At the top, there's a navigation bar with links for 'Default', 'STACKS', 'CATALOG', 'INFRASTRUCTURE', 'ADMIN' (with a red exclamation mark), and 'API'. Below the navigation is a teal header bar with a warning icon and the text: 'Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host'. The main content area has a title 'User Stacks' and a button 'Add Stack'. On the right, there's a 'Sort By:' dropdown with options 'State' and 'Name'. A large central box is titled 'Adding your first Stack'. It contains text explaining that a service is a group of containers from the same Docker image with extended features like DNS discovery, and that it can be added individually or from the Catalog. It also mentions built-in services like load balancers and monitoring. Below this text are two buttons: 'Define a Service' and 'Browse Catalog'. At the bottom of the page, there's a footer with links for 'v1.6.17', 'Help', 'Documentation', 'File an Issue', 'Forums', 'Slack', 'English' (with a dropdown arrow), and 'Download CLI'.

v1.6.17 Help Documentation File an Issue Forums Slack

English ▾ Download CLI ▾

0.0.0.0:8080

Lunch

Schedule

- What DevOps is
- What is CI/CD
- Docker
- Docker-compose
- Setup your MySql cluster for high availbailtiy (MariaDB Galera Cluster)
- Rancher
- CircleCI
- ELK for Logging

MariaDB Galera Cluster

MariaDB Galera Cluster is a synchronous **multi-master cluster** for MariaDB. It is available on Linux only, and only supports the XtraDB/InnoDB storage

MariaDB Galera Cluster

- Synchronous replication
- Active-active multi-master topology
- Read and write to any cluster node
- Automatic membership control, failed nodes drop from the cluster
- Automatic node joining
- True parallel replication, on row level
- Direct client connections, native MariaDB look & feel

MariaDB Galera Cluster

- at least 3 node per cluster (recommend 5)
- memory at least 1Gb per node
- private networking if possible

MariaDB Galera Cluster

Run setup.sh

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv BC19DDBA
sudo add-apt-repository 'deb http://releases.galeracluster.com/mysql-wsrep-5.6/ubuntu xenial main'
sudo add-apt-repository 'deb http://releases.galeracluster.com/galera-3/ubuntu xenial main'
sudo echo "Package: *" >> /etc/apt/preferences.d/galera.pref
sudo echo "Pin: origin http://releases.galeracluster.com" >> /etc/apt/preferences.d/galera.pref
sudo echo "Pin-Priority: 1001" >> /etc/apt/preferences.d/galera.pref
sudo apt-get update
sudo apt-get install galera-3 galera-arbitrator-3 mysql-wsrep-5.6 -y
sudo apt-get install rsync -y
mysql -furoot < "mysql_secure_installation.sql"
sudo systemctl stop mysql
```

MariaDB Galera Cluster

Same step for first node

- sudo systemctl stop mysql

MariaDB Galera Cluster

Same step for every node

- sudo nano /etc/mysql/conf.d/galera.cnf

```
/etc/mysql/conf.d/galera.cnf on the first node

[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so

# Galera Cluster Configuration
wsrep_cluster_name="test_cluster"
wsrep_cluster_address="gcomm://first_ip,second_ip,third_ip"

# Galera Synchronization Configuration
wsrep_sst_method=rsync

# Galera Node Configuration
wsrep_node_address="this_node_ip"
wsrep_node_name="this_node_name"
```

MariaDB Galera Cluster

Same step for every node

/etc/mysql/conf.d/galera.cnf on the first node

```
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so

# Galera Cluster Configuration
wsrep_cluster_name="test_cluster"
wsrep_cluster_address="gcomm://first_ip,second_ip,third_ip"

# Galera Synchronization Configuration
wsrep_sst_method=rsync

# Galera Node Configuration
wsrep_node_address="this_node_ip"
wsrep_node_name="this_node_name"
```

MariaDB Galera Cluster

First node on Cluster

- sudo /etc/init.d/mysql start --wsrep-new-cluster
- sudo galera_new_cluster
- tail -f /var/log/syslog
- mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"

other node on cluster

- sudo systemctl start mysql
- mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
-

Workshop: setup Galera

Tip: MariaDB Galera Cluster

```
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

max_connections=1024
key_buffer_size= 20M
table_open_cache = 256
max_allowed_packet = 32M
sort_buffer_size = 20M
read_buffer_size = 20M
thread_concurrency = 4

# INNODB #
innodb-flush-method          = O_DIRECT
innodb-log-files-in-group    = 2
innodb-log-file-size         = 512M
innodb-flush-log-at-trx-commit = 1
innodb-file-per-table        = 1
innodb-buffer-pool-size      = 2G
```

Tip: MariaDB Galera Cluster

```
# change auto_increment_increment and auto_increment_offset automatically
wsrep_auto_increment_control=1

# enable "strictly synchronous" semantics for read operations
wsrep_causal_reads=0

# retry autoinc insert, which failed for duplicate key error
wsrep_drupal_282555_workaround=1

wsrep_retry_autocommit=3
```

MariaDB Galera Cluster: รับมือกับพญานาค

Can't Auth after grant user access

MariaDB Galera Cluster: Reset password

```
/etc/init.d/mysql stop
```

```
mysqld_safe --skip-grant-tables &
```

```
mysql -uroot
```

```
use mysql;
```

```
flush privileges;
```

```
select * from user;
```

```
UPDATE user SET Password =  
PASSWORD('thisispassword')  
WHERE User = 'user_a';
```

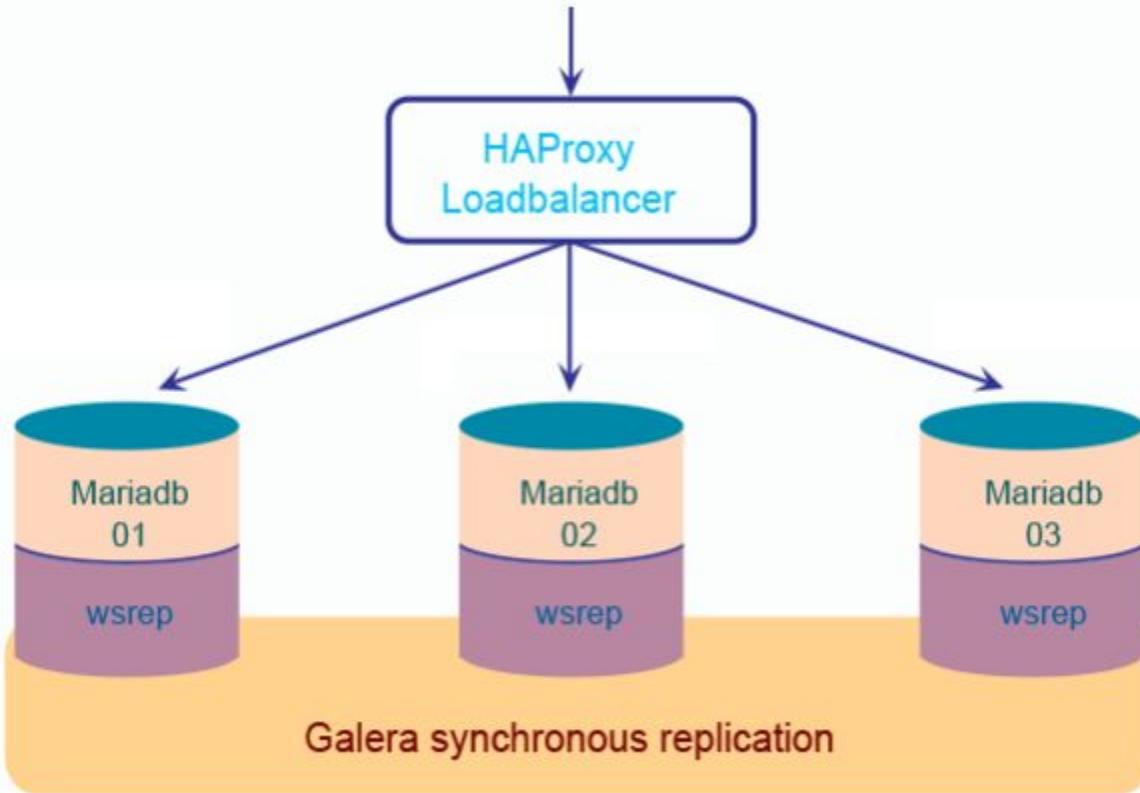
```
GRANT ALL PRIVILEGES ON *.* TO  
'user_a' WITH GRANT OPTION;
```

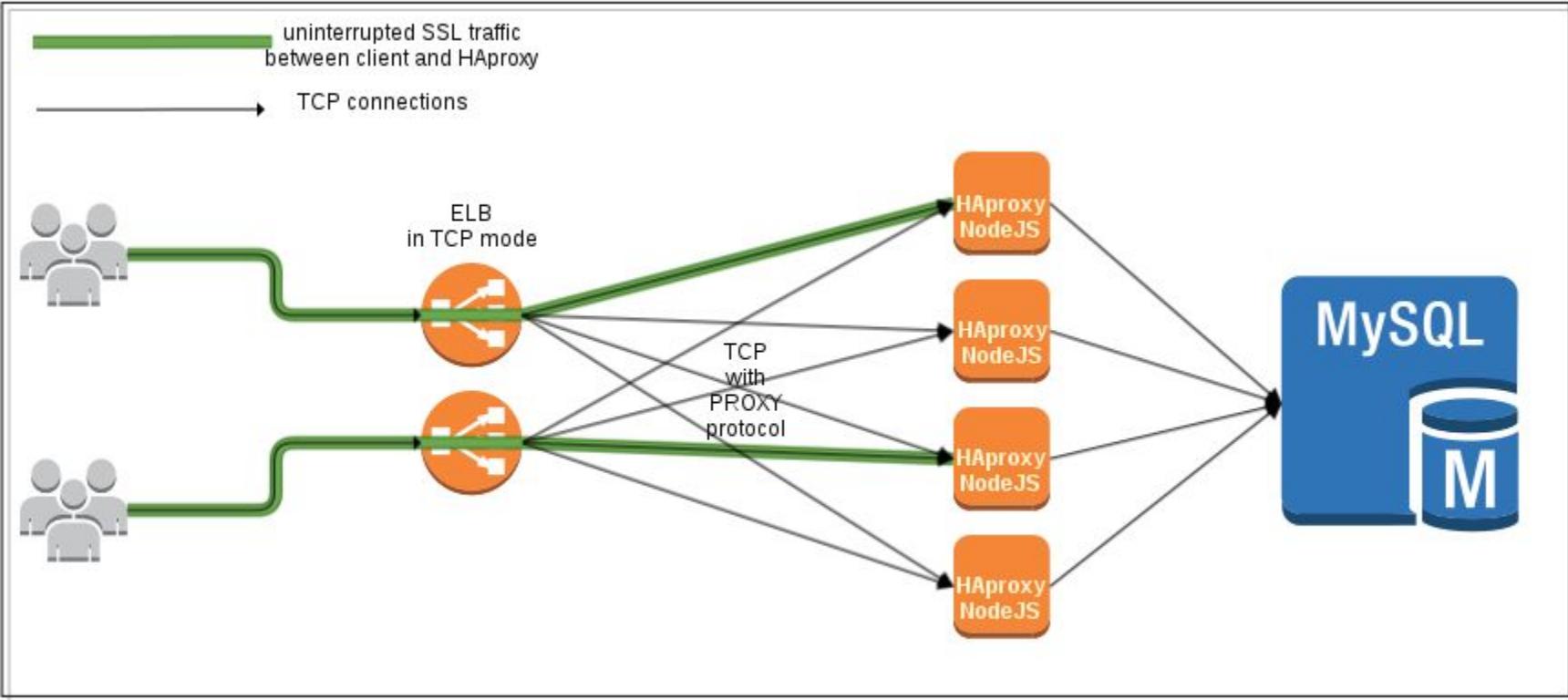
```
flush privileges;
```

```
systemctl start mysql
```

MariaDB Galera Cluster: Reset password

repeat every node in cluster





Load balance Galera Cluster with NGINX

```
stream {
    upstream galera {
        least_conn;
        server 10.1.1.10:3306;
        server 10.1.1.20:3306;
        server 10.1.1.30:3306;
        zone tcp_mem 64k;
    }

    server {
        listen 13306;
        proxy_pass galera;
        proxy_connect_timeout 1s;
    }
}
```

Recovery Galera Cluster

```
$ cat /var/lib/mysql/grastate.dat
```

```
# GALERA saved state
version: 2.1
uuid:      38869bf8-bd84-11e8-b76a-7f8d264cd8bf
seqno:    -1
safe_to_bootstrap: 0
```

Schedule

- What DevOps is
- What is CI/CD
- Docker
- Docker-compose
- Setup your MySql cluster for high availbailty (MariaDB Galera Cluster)
- Rancher
- CircleCI
- ELK for Logging

4 Steps for make your pipeline

1. Prepare your Rancher server
2. Add new server (worker)
3. Dockerfile
4. Circleci config

Prepare your Rancher server

1. Rancher node
2. Database node

Workshop: rancher II

“ setup rancher with Galera Cluster“

4 Steps for make your pipeline

1. Prepare your Rancher server
2. Add new server (worker)
3. Dockerfile
4. Circleci config

Rancher first time

1. Set Access Control!!!

Before adding your first service or launching a container, you'll need to a reported version of Docker. Add a host

Access Control

Active Directory

GITHUB

LOCAL

OpenLDAP

SHIBBOLETH

GitHub is not configured

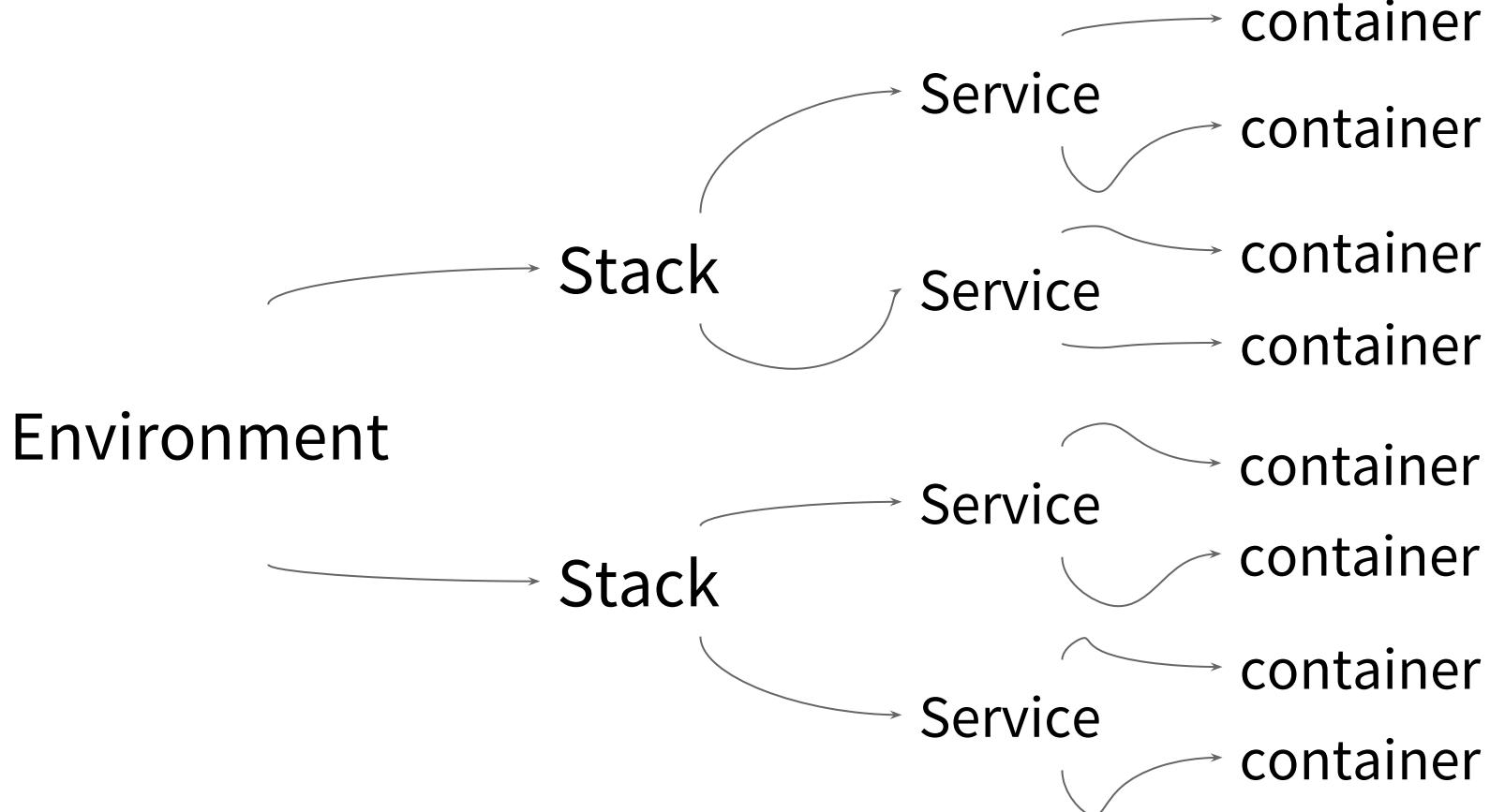
Rancher can be configured to restrict access to a set of GitHub users and organization members. This is not currently set up, so anybody that reach this page (or the API) has full control over the system.

1. Setup a GitHub Application

- For standard GitHub, click [here](#) to go applications settings in a new window.
For Github Enterprise, login to your account. Click on Settings, then Applications.
- Click "Register new application" and fill out the form:
Application name: Anything you like, e.g. My Rancher

Rancher first time

1. Environment
2. Stack
3. Service



Rancher first time

1. Environment

Manage Environments

Name	Description
e.g. lab	e.g. Environment for developer experimentation

Environment Template


Cattle


Kubernetes


Mesos


Swarm
EXPERIMENTAL


Windows
EXPERIMENTAL

Orchestration: Cattle
Framework: Network Services, Scheduler, Healthcheck Service
Networking: Rancher IPsec

Rancher first time

2. Stack

Add Stack

Name	Description
e.g. myapp	e.g. MyApp Stack

OPTIONAL: IMPORT COMPOSE

Optional: docker-compose.yml	Optional: rancher-compose.yml
Contents of docker-compose.yml	Contents of rancher-compose.yml

[ADVANCED OPTIONS ^](#)

Rancher first time

3. Service

Stack:		Staging	Add Service	Active	⋮
Active	after5-api-staging ⓘ	Image: af... ./after5-api-staging:latest	Service	1 Container	⋮
Active	after5-app-staging ⓘ	Image: af... ./after5-app-staging:latest	Service	1 Container	⋮
Active	after5-driver-staging ⓘ	Image: af... ./after5-driver-staging:latest	Service	1 Container	⋮
Active	after5-manage-staging ⓘ	Image: af... ./after5-manage:latest	Service	1 Container	⋮

2. Add new server (worker)

1. Get access_token
2. Click Click Click



Manage available machine drivers

ACCOUNT ACCESS

Access Token*

Your DigitalOcean API access token

A Personal Access Token from the DigitalOcean Apps & API screen

Next: Configure Droplet

Cancel

Demo: rancher

“ Create new server “

Workshop: rancher III

“ Create new 2 server for worker “

Manual run rancher client

```
crontab -e
```

```
@reboot sudo docker run --rm --privileged -v /var/run/docker.sock:/var/run/docker.sock -v  
/var/lib/rancher:/var/lib/rancher rancher/agent:v1.2.10  
http://111.222.333.444:8080/v1/scripts/ABCDDFDFFFEFEFEFBE0DDCD65C:15131231238400000:  
jhYGgwgwgb5Jh1pKeB9G7KkEhi0g
```

Demo: rancher I

“ Deploy example app “

Demo: rancher I

- image: nginx
- port 80

Workshop: rancher IV

“ Deploy your own image with rancher “

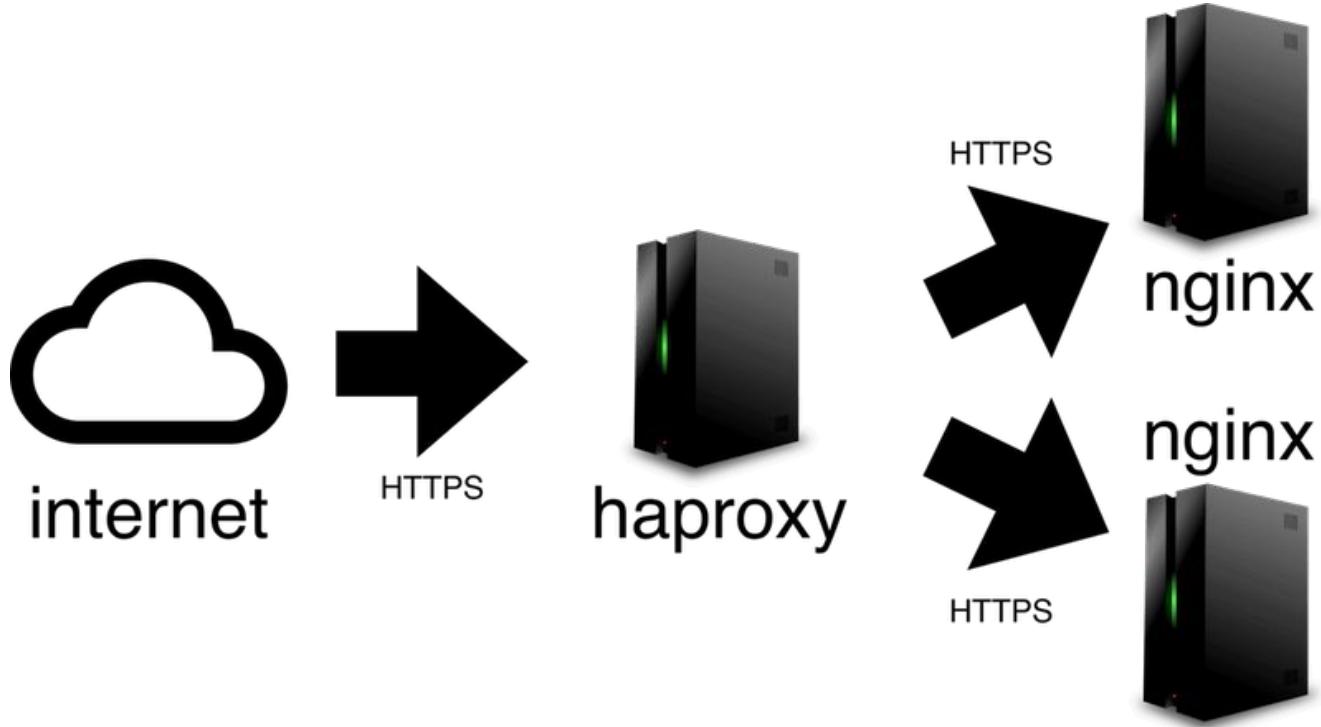
Workshop: rancher V

“ Deploy your own docker-compose with rancher “

Workshop: rancher VI

“ Deploy Mutil Containner“

Load Balancing with Haproxy

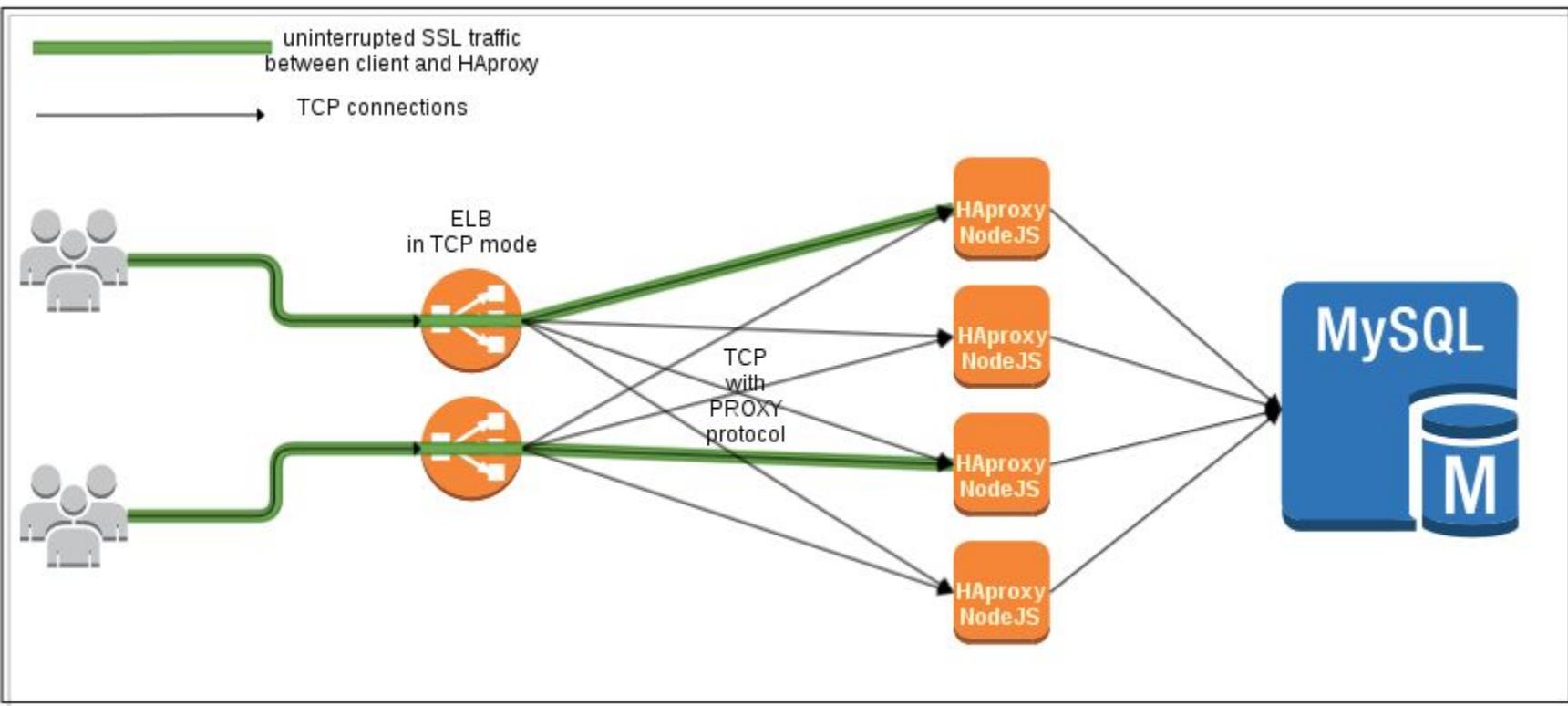


Demo: Load Balancing

“ Create load balancing for example “

Workshop: Load Balancing

“ Create load balancing for your app “



4 Steps for make your pipeline

1. Prepare your Rancher server
2. Add new server (worker)
3. Dockerfile
4. Circleci config

Break

circleci

Create a Pipeline for Deploying

circleci configuration

```
version: 2
jobs:
  build:
    docker:
      - image: circleci/node:8.11.1-stretch
    working_directory: ~/repo
    branches:
      only:
        - master
        - staging
    steps:
      - checkout
```

circleci configuration

```
version: 2
jobs:
  build:
    docker:
      - image: circleci/node
    working_directory: ~/repo
    branches:
      only:
        - master
        - staging
    steps:
      - checkout
```

Build: Job name

- At least it should be job name ‘build’

docker:

image:

- tell circleci to pull docker image for this ENV.

branches:

- Select branches to action/build

steps:

- what you want circleci to do

circleci configuration

```
steps:
  - checkout
  - run: yarn install
  - setup_remote_docker
  - run: docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
  - run:
      name: "Build docker image and push to docker hub"
      command: |
        docker build -t ima8/codemania111:latest .
        docker push ima8/codemania111:latest
```

Demo: Circleci

“ Overview Circleci and import project “



Authorize CircleCI



CircleCI by [circleci](#)

wants to access your `gasileer` account



Personal user data

Email addresses (read-only)



Repositories

Public and private



[Authorize circleci](#)

Authorizing will redirect to

<https://circleci.com>



Not owned or
operated by GitHub



Created 7 years ago



More than 1K
GitHub users

[Learn more about OAuth](#)

[Updates](#)[Support](#)

JOBS



WORKFLOWS



INSIGHTS

ADD
PROJECTS

TEAM



SETTINGS



Welcome to CircleCI!

You've joined the ranks of 100,000+ teams who ship better code, faster.

Getting Started

On this page, choose projects to populate your dashboard from the ones that are **already building** on CircleCI.

In the next view, you'll be able to **set up new projects**. If you don't want to follow any of the projects listed below, or haven't set up any projects with CircleCI before, you can [skip ahead to set up new projects](#).

»  **gasileer** No projects building on CircleCI. Skip to set up new projects.

[Add Projects](#)

Interested in a tour?

[See how Spotify uses CircleCI](#). You'll be able to see what builds pass/fail and show how fast they run.

Demo: Circleci II

“ Run the first build: just npm install “

Workshop: CircleCI

“Link your github and Run the first build”

workshop: CircleCI II

“ Build docker and push to docker hub “

```
version: 2
jobs:
  build:
    docker:
      # specify the version you desire here
      - image: circleci/node:8.11.1-stretch
    working_directory: ~/repo
    steps:
      - checkout
      - restore_cache:
          keys:
            - v1-dependencies-{{ checksum "package.json" }}
            # fallback to using the latest cache if no exact match is found
            - v1-dependencies-
      - run: npm install
      - save_cache:
          paths:
            - node_modules
          key: v1-dependencies-{{ checksum "package.json" }}
      - run: npm test
      - setup_remote_docker
      - run: docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
      - run:
          name: "Build docker image and push to docker hub"
          command: |
            cp Dockerfile.production Dockerfile
            docker build -t ima8/example-nodejs-circle:latest .
            docker build -t ima8/example-nodejs-circle:${CIRCLE_SHA1} .
            docker push ima8/example-nodejs-circle:latest
            docker push ima8/example-nodejs-circle:${CIRCLE_SHA1}
      - run: echo Done
```

circleci configuration

<https://github.com/etlweather/gaucho>

The screenshot shows the GitHub repository page for `etlweather / gaucho`. The repository description is "A Python CLI tool for Rancher's API". It has 64 commits, 1 branch, 0 releases, and 11 contributors. The latest commit is 80b4882 from 17 Apr. The repository includes files like `.gitignore`, `Docker.alpine`, `Dockerfile`, `LICENSE`, and `README.md`.

A Python CLI tool for Rancher's API

rancher rancher-api ci-cd deployment

64 commits 1 branch 0 releases 11 contributors MIT

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

etlweather Merge pull request #30 from nvgoldin/add_requirements_file ... Latest commit 80b4882 on 17 Apr

File	Description	Time
<code>.gitignore</code>	remove and deactivate environment	5 months ago
<code>Docker.alpine</code>	Alpine-based Docker image.	3 months ago
<code>Dockerfile</code>	improved Dockerfile	7 months ago
<code>LICENSE</code>	Basic setup of repository.	2 years ago
<code>README.md</code>	id_of_env	5 months ago

circleci configuration

```
- run:  
  name: "Call to rancher to deploy"  
  command: |  
    docker run --rm -it \  
      -e CATTLE_ACCESS_KEY="$CATTLE_ACCESS_KEY" \  
      -e CATTLE_SECRET_KEY="$CATTLE_SECRET_KEY" \  
      -e CATTLE_URL="$CATTLE_URL" \  
      etlweather/gaucho upgrade $RANCHER_EXAMPLE_NODEJS \  
      --imageUuid 'docker:ima8/example-nodejs-circle:latest' \  
      --batch_size 3 --start_first \  
      --auto_complete --timeout 600 \  
  /
```

Workshop: CircleCI III

“ Deploy your app to rancher“

circleci configuration: cache

```
steps:
  - checkout
  - restore_cache:
      keys:
        - v1-dependencies-{{ checksum "package.json" }}
        - v1-dependencies-
  - run: cd example_worker
  - run: yarn install
  - save_cache:
      paths:
        - node_modules
    key: v1-dependencies-{{ checksum "package.json" }}
```

Workshop: CircleCI IV

“ Deploy your app to rancher with cache“

circleci with production and staging

```
- setup_remote_docker
- run: docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
- run:
  name: "Build docker image and push to docker hub"
  command: |
    if [ "${CIRCLE_BRANCH}" == "master" ]; then
      docker build -t ima8/code_mania_demo:latest .
      docker build -t ima8/code_mania_demo:${CIRCLE_SHA1} .
      docker push ima8/code_mania_demo:latest
      docker push ima8/code_mania_demo:${CIRCLE_SHA1}
    elif [ "${CIRCLE_BRANCH}" == "staging" ]; then
      docker build -t ima8/code_mania_demo_staging:latest .
      docker push ima8/code_mania_demo_staging:latest
    else
      echo "This is ${CIRCLE_BRANCH}"
    fi
```

Workshop: Circleci V

“ Deploy your app to rancher with selected branch“

Workshop: CircleCI VI

“ Add run test before deploy ”

Day 2

Workshop: CircleCI VII

```
{  
  env: "staging",  
  uptime: "00:00:19",  
  + networkInterfaces: {...}  
}
```

“ Deploy your app end to end from git to ranche with
5 containers in 2 servers “

<https://github.com/lma8/example-nodejs-express>

circleci workflows

```
jobs:  
  build: ...  
  test:  
  docker:  
    - image: circleci/node:8.11.1-stretch  
  working_directory: ~/repo  
  steps:  
    - checkout  
    - restore_cache:  
        keys:  
          - v1-dependencies-{{ checksum "package.json" }}  
          - v1-dependencies-  
    - run: npm install  
    - save_cache:  
        paths:  
          - node_modules  
        key: v1-dependencies-{{ checksum "package.json" }}  
    - run: npm test  
ship: ...  
deploy: ...  
workflows:  
  .
```

```
workflows:  
  version: 2  
  build_test_ship_deploy:  
    jobs:  
      - build  
      - test:  
          requires:  
            - build  
      - ship:  
          requires:  
            - build  
            - test  
      - deploy:  
          type: approval  
          requires:  
            - build  
            - test  
            - ship
```

Workflows » lma8 » example-nodejs-express-circleci » master » 60590763-f20b-4316-8fc8-17f7fa711e04

ON HOLD

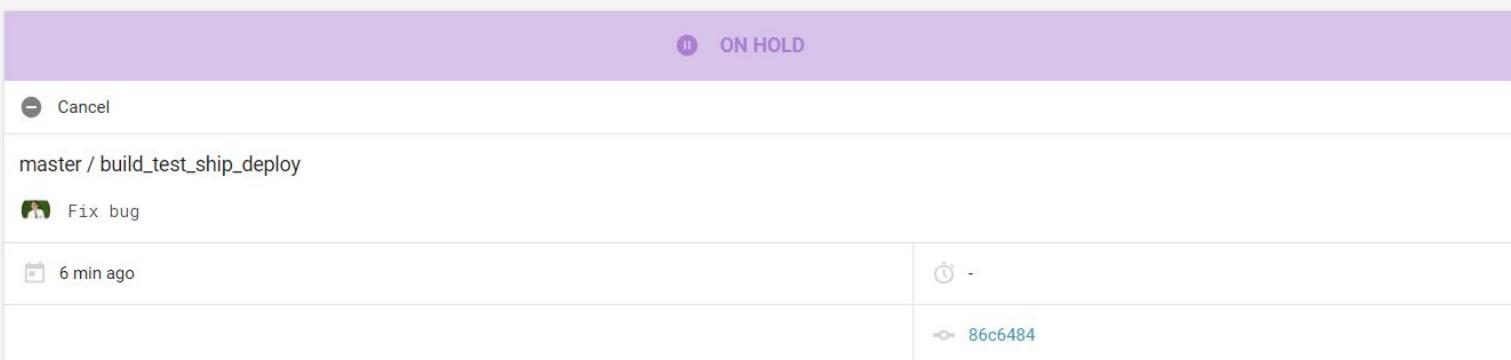
Cancel

master / build_test_ship_deploy

Fix bug

6 min ago

86c6484



4 jobs in this workflow



ON HOLD

Cancel

master / build_test_ship_deploy

Fix bug

6 min ago

4 jobs in this workflow

build 00:09 → test 00:23 → ship 00:50 → deploy

Approve Job

To continue running this workflow, click Approve.

Cancel Approve

The image shows a screenshot of a CircleCI workflow interface. At the top, it says 'ON HOLD'. Below that is a 'Cancel' button. The main title is 'master / build_test_ship_deploy'. Underneath, there's a 'Fix bug' entry with a timestamp of '6 min ago'. A message at the bottom left says '4 jobs in this workflow'. At the bottom, a horizontal sequence of four boxes represents the workflow: 'build' (green checkmark), 'test' (green checkmark), 'ship' (green checkmark), and 'deploy' (purple double-dot icon). Above this sequence is a green arrow pointing from 'build' to 'test', another green arrow from 'test' to 'ship', and a purple arrow from 'ship' to 'deploy'. A prominent white modal box is centered over the workflow. It has a title 'Approve Job' and a message 'To continue running this workflow, click Approve.' At the bottom of the modal are two buttons: 'Cancel' (gray) and 'Approve' (blue).

Workshop: Circleci VIII

“ Deploy your app end to end from git to ranche with
5 containers in 2 servers **with workflow**“

<https://github.com/lma8/example-nodejs-express>

Tip: Circleci for fast deploy

พยายามย้าย Command ไปรันข้างนอก Docker ให้มากที่สุด

เช่น

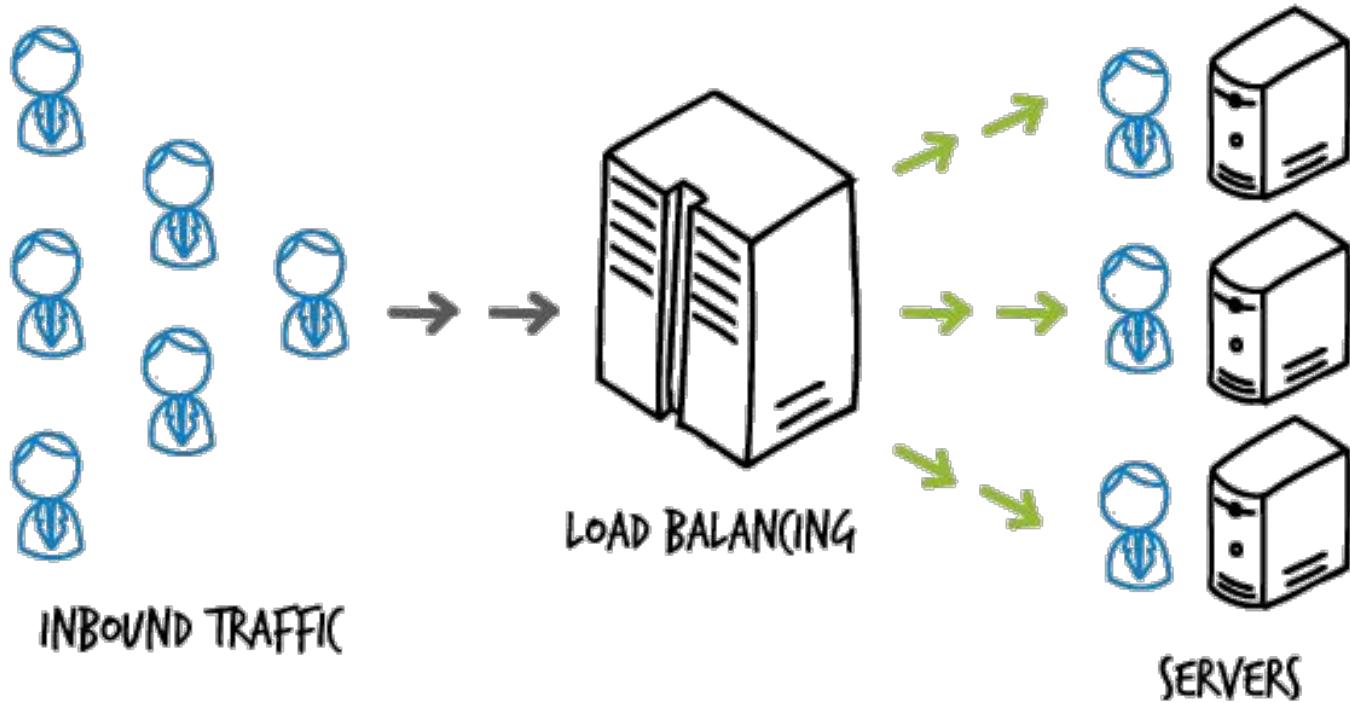
- npm install
- npm run test

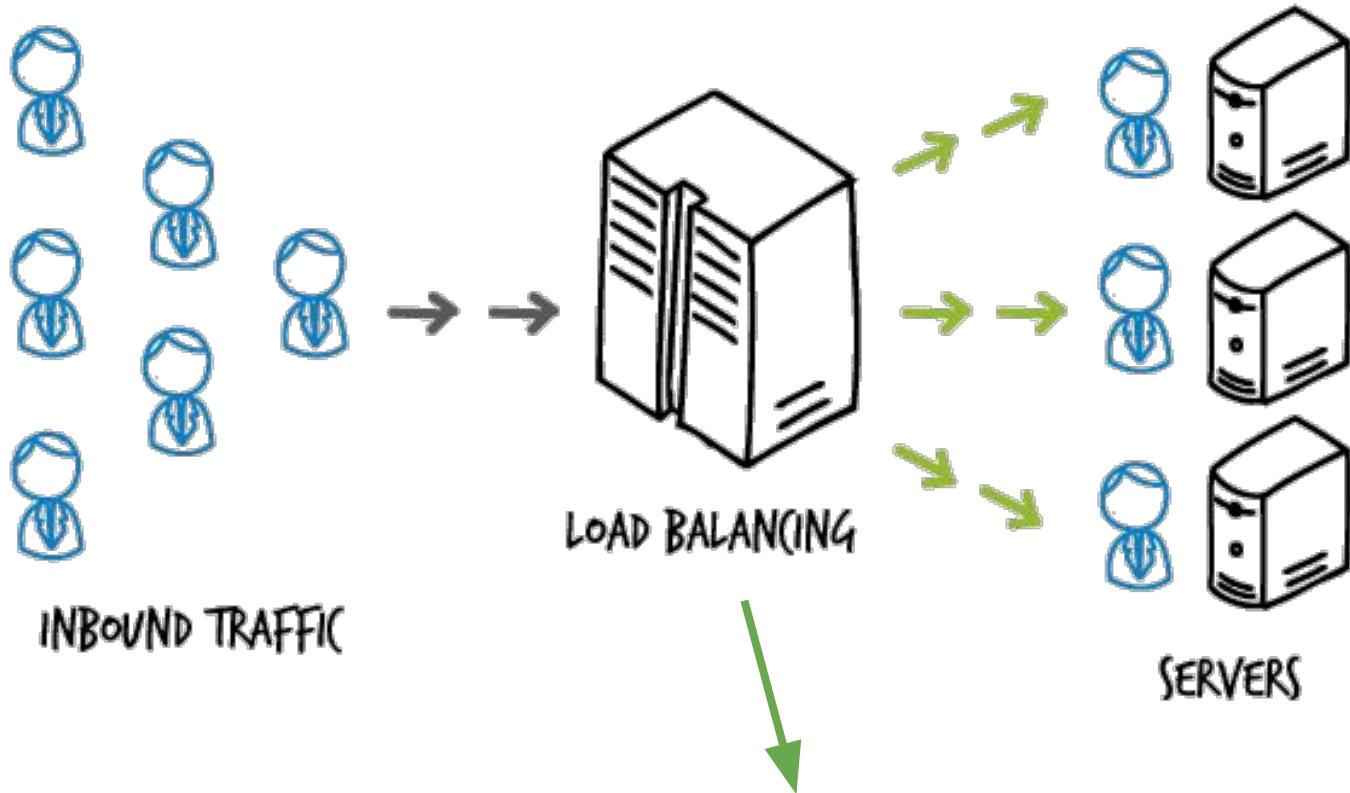
ELK

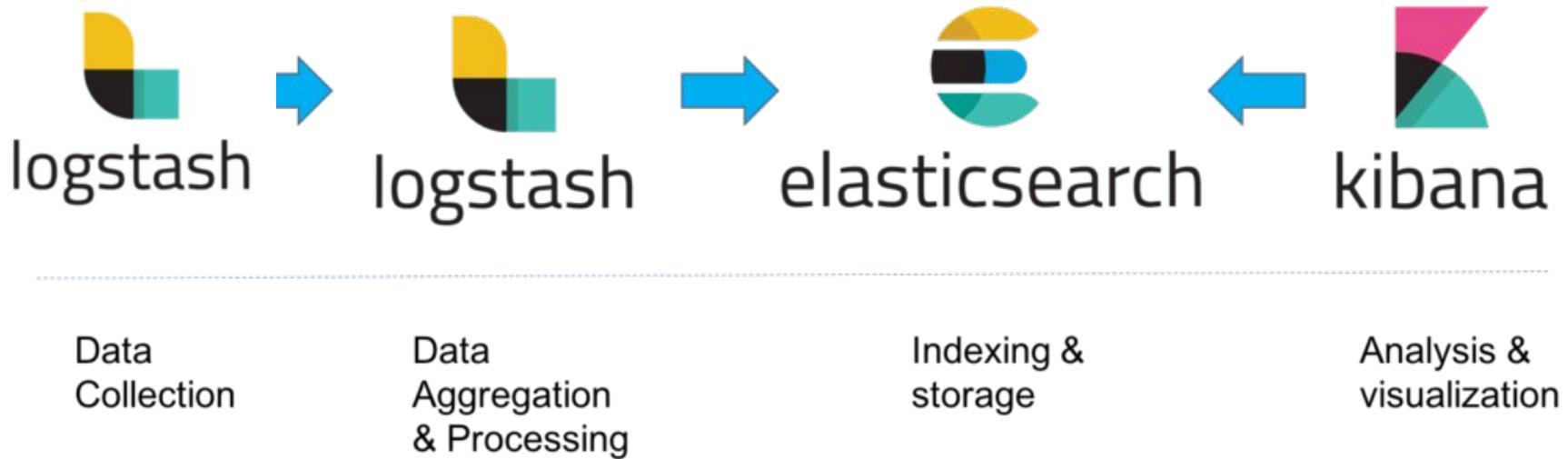
- **Elasticsearch:** is a full-text distributed NoSQL database
- **Kibana:** is a window into the Elastic Stack
 - i.e. Dashboard
- **Logstash:** is a data collection enginee and data processing pipeline

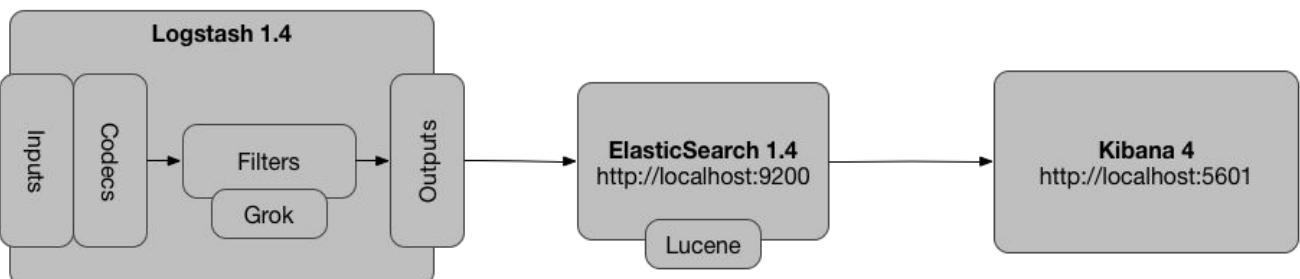
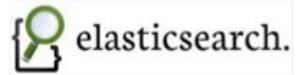
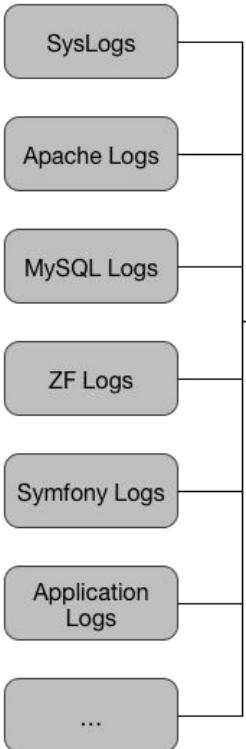
ELK

centralized logging









The **ELK** platform for Log management



Setup Elasticsearch with high availability

After5-dev ▾ STACKS ▾ CATALOG ▾ INFRASTRUCTURE ▾ ADMIN ▾ API ▾

Catalog: All Category: All Manage

 Alfresco An ECM and BPM platform. View Details	 Alibaba Cloud DNS Rancher External DNS service powered by Alibaba Cloud View Details	 Apache Guacamole Apache Guacamole is a clientless remote desktop gateway. It supports standard protocols like VNC, RDP, and SSH. View Details	 Apache Kafka Kafka cluster View Details	 Apache Zookeeper Zookeeper cluster View Details	 Aqua Aqua Container Security Platform View Details
 Artifactory Artifactory is a universal Binary Repository Manager View Details	 ascinema.org ascinema.org is a free and open source solution for recording terminal sessions and sharing them on the web. View Details	 AutoSpotting Replaces AWS On-Demand instances with cheaper Spot-Instances where possible View Details	 Avi Vantage Platform External LB service powered by Avi Vantage Platform View Details	 AWS Spot Instance Helper Automatically evicts spot instances that are marked for termination View Details	 AWX AWX provides a web-based user interface, REST API, and task engine built on top of Ansible. It is the View Details

Setup Elasticsearch with high availability

Catalog: All

 Elasticsearch

Category: All 

 Manage



Elasticsearch

Elasticsearch, you know for search!

[View Details](#)



Elasticsearch 2.x

Elasticsearch, you know for search!

[View Details](#)



Elasticsearch Cluster 6.2.4

Elasticsearch, you know for search!

[View Details](#)



PHP Adminer

Adminer (formerly phpMinAdmin) is a full-featured database management tool written in PHP. Conversely to phpMyAdmin, it consist of a single ...

[View Details](#)

Name*

es-cluster

Description

Description

Configuration Options

Cluster name*

es-cluster

Name of the Elasticsearch Cluster

Heap size (master nodes)*

512m

Heap size to be allocated for Java (master nodes)

Heap size (data nodes)*

512m

Heap size to be allocated for Java (data nodes)

Heap size (client nodes)*

512m

Heap size to be allocated for Java (client nodes)

of minimum Master Nodes*

3

Update host sysctl:*

false

Set true to avoid vm.max_map_count errors. WARN: If set true, host param vm.max_map_count will be update to 262144.

Memory limit in byte (master nodes)*

1073741824

Memory limit in Byte per elasticsearch container. AT LEAST double the heap size! (master nodes)

Memory limit in byte (data nodes)*

1073741824

Memory limit in Byte per elasticsearch container. AT LEAST double the heap size! (data nodes)

Memory limit in byte (client nodes)*

1073741824

Memory limit in Byte per elasticsearch container. AT LEAST double the heap size! (client nodes)

of initial data nodes*

2

Setup Elasticsearch with high availability

- Master node: controls the cluster
- Data node: hold data and perform data related operations such as CRUD, search, and aggregations.
- Client node(Ingest node) : smart load balancers

Setup Elasticsearch with high availability

<https://www.elastic.co/videos/distributed-diagram>

Setup Elasticsearch with high availability

Cluster name*

es-cluster

Name of the Elasticsearch Cluster

Heap size (master nodes)*

1024m

X2

Heap size to be allocated for Java (master nodes)

Heap size (data nodes)*

6114m

X2

Heap size to be allocated for Java (data nodes)

Heap size (client nodes)*

1024m

X2

Heap size to be allocated for Java (client nodes)

of minimum Master Nodes*

3

Set the number of required master nodes to reach quorum. Sets initial scale to this value as well

of initial client nodes*

1

Set the initial number of client nodes

Update host sysctl:*

true

Set true to avoid vm.max_map_count errors. WARN: If set true, host param vm.max_map_count will be update to 262144.

Memory limit in byte (master nodes)*

2000000000

Memory limit in Byte per elasticsearch container. AT LEAST double the heap size! (master nodes)

Memory limit in byte (data nodes)*

12000000000

Memory limit in Byte per elasticsearch container. AT LEAST double the heap size! (data nodes)

Memory limit in byte (client nodes)*

2000000000

Memory limit in Byte per elasticsearch container. AT LEAST double the heap size! (client nodes)

of initial data nodes*

2

Set the initial number of data nodes

VOLUME Driver*

local

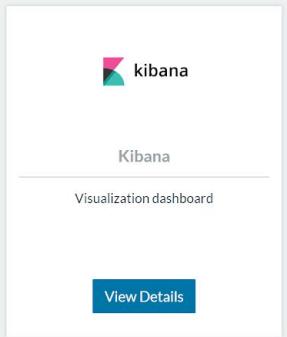
172*

The VOLUME driver to associate with this server

Setup Kibana

Catalog: All

kibana Category: All Manage



Setup Kibana

Template Version

6.2.3-rancher1

Select a version of the template to deploy

New Stack

Name*

kibana

Description

Description

Configuration Options

Elasticsearch source*

es-cluster/es-client

[Link to elasticsearch service or stack/service](#)

Public Port*

80

[Unique public port for Kibana](#)

Start services after creating

Setup Kibana

Template Version

6.2.3-rancher1

Select a version of the template to deploy

New Stack

Name*

kibana

Description

Description

Configuration Options

Elasticsearch source*

es-cluster/es-client

[Link to elasticsearch service or stack/service](#)

Public Port*

80

Unique public port for Kibana

Start services after creating

PREVIEW ▾

Setup Kibana



Container Overview						Actions
State	Name	IP Address	Host	Image	Stats	Action
Running	kibana-nginx-proxy-1	10.42.229.95	Data-collection-02	rancher/nginx:v1.9.4-3	 	 
Running	kibana-nginx-proxy-kibana6-1	None	Data-collection-02	docker.elastic.co/kibana/kibana:6...	 	 
Running	kibana-nginx-proxy-nginx-proxy...	10.42.41.44	Data-collection-02	rancher/nginx-conf:v0.2.0	 	 

Setup Kibana

kibana:6.2.4

Upgrade 3 Services

Batch Size

1

Batch Interval

2

Start Behavior

Start before Stopping

Which Services

- nginx-proxy
- kibana6
- nginx-proxy-conf

kibana6

Select Image*

docker.elastic.co/kibana/kibana:6.2.4

Always pull image before creating

 Port Map

Hello Kibana

kibana

- Discover
- Visualize
- Dashboard
- Timelion
- APM
- Dev Tools
- Monitoring
- Management

Add Data to Kibana

Use these solutions to quickly turn your data into pre-built dashboards and monitoring systems.

Data already in Elasticsearch?

[Set up index patterns](#)



APM

APM automatically collects in-depth performance metrics and errors from inside your applications.

[Add APM](#)



Logging

Ingest logs from popular data sources and easily visualize in preconfigured dashboards.

[Add log data](#)



Metrics

Collect metrics from the operating system and services running on your servers.

[Add metric data](#)



Security analytics

Centralize security events for interactive investigation in ready-to-go visualizations.

[Add security events](#)

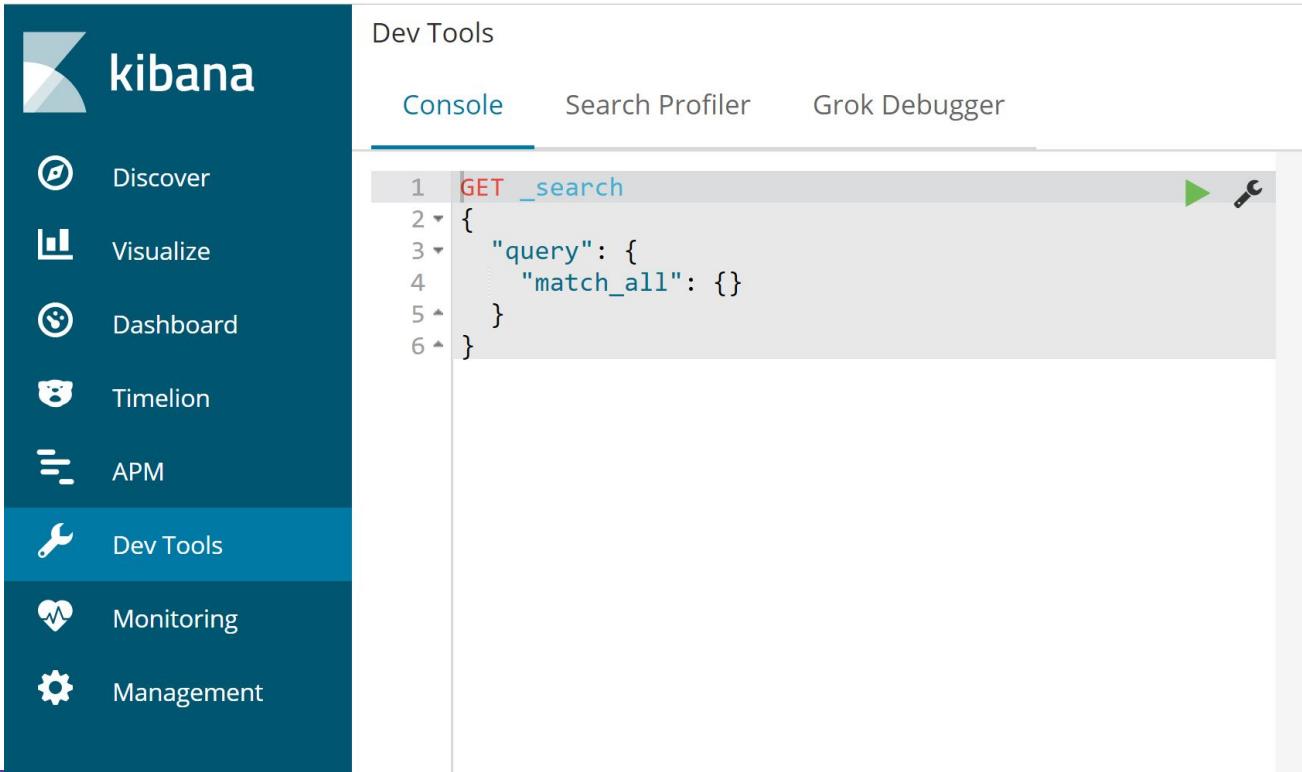
[Visualize and Explore Data](#)

[Manage and Administer the Elastic Stack](#)

Workshop: ELK I

“ Setup your Elasticsearch and Kibana “

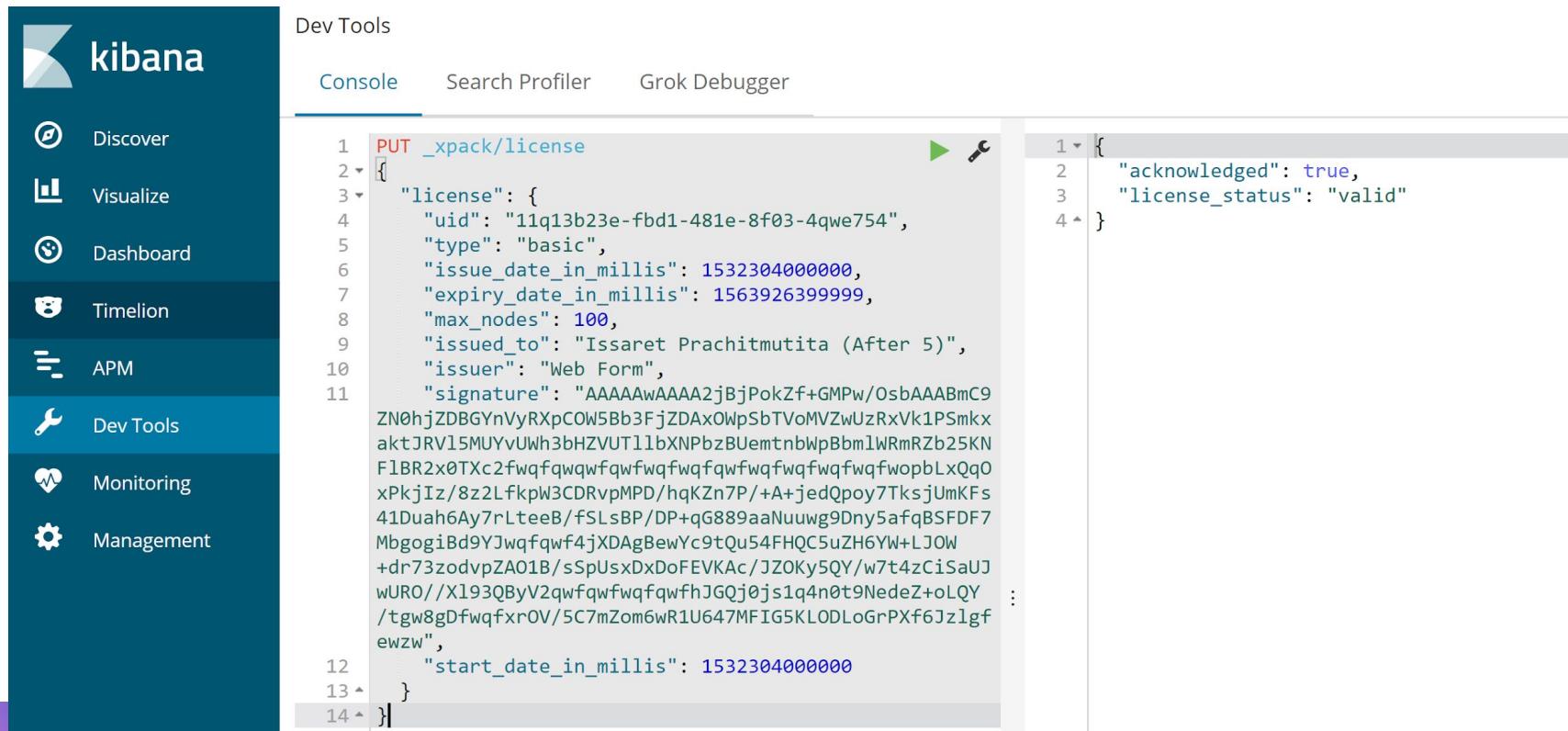
Elasticsearch first time



The screenshot shows the Kibana interface with the Dev Tools section active. The left sidebar lists various features: Discover, Visualize, Dashboard, Timelion, APM, Dev Tools (which is selected and highlighted in blue), Monitoring, and Management. The main area is titled "Dev Tools" and contains three tabs: Console, Search Profiler, and Grok Debugger. The "Console" tab is selected and displays a code editor with a query. The query is a single-line GET request to the _search endpoint with a match_all query. The code editor has line numbers 1 through 6 and a green run button and a wrench icon.

```
1 GET _search
2 {
3   "query": {
4     "match_all": {}
5   }
6 }
```

Install license



The screenshot shows the Kibana interface with the Dev Tools panel open. The left sidebar includes links for Discover, Visualize, Dashboard, Timelion, APM, Dev Tools (which is selected), Monitoring, and Management. The Dev Tools panel has tabs for Console, Search Profiler, and Grok Debugger. The Console tab is active, displaying a command-line interface with syntax highlighting for JSON and code snippets.

The command entered in the console is:

```
PUT _xpack/license
```

The response shown on the right is:

```
{  
  "acknowledged": true,  
  "license_status": "valid"
```

What is ELASTICSEARCH

A real-time distributed **search** and
analytic engine

What is Elasticsearch abilities?

- **Full-text** search
- **Structured** search
- Real-time **analytics**
- Combination all of the above

Elasticsearch features

- **Distributed document store:**
 - RESTful API
 - Automatic scale
 - Plug & Play

Elasticsearch features

- Handle the **human language**:
 - Score results by **relevance**
 - **Synonyms**
 - **Typos and misspellings**
 - Internationalization

Elasticsearch features

- Powerful analytics:
 - Comprehensive **aggregations**
 - Geolocations
 - Can be combined with **search**
 - **Real-time** (no batch-processing)

Elasticsearch features

- **Free and open source**
- Community support
- Backed by Elasic Co.

Compare with SQL Databases

SQL = Structured Query Language

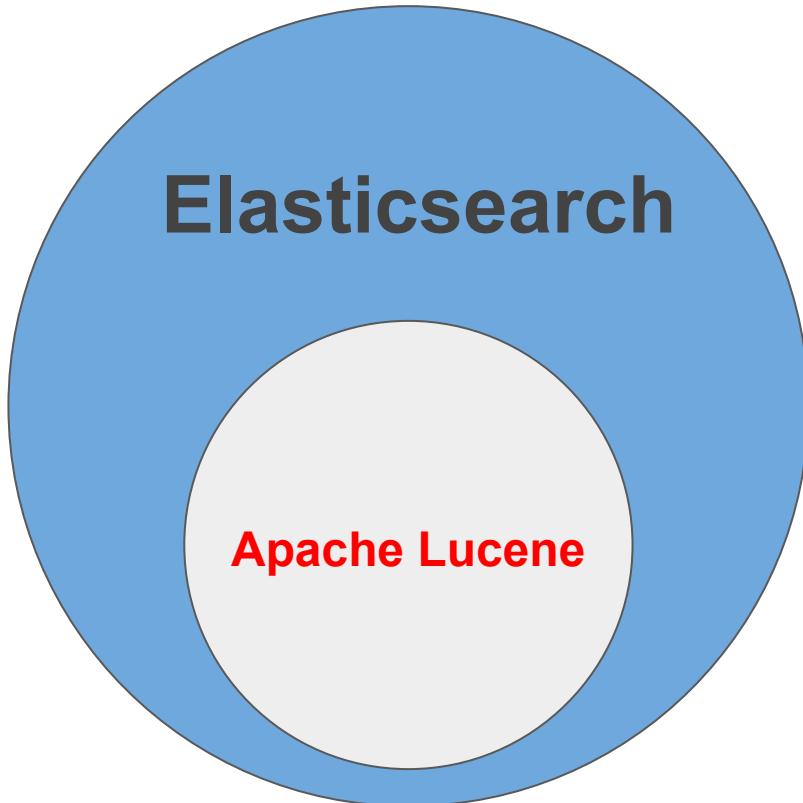
Compare with SQL Databases

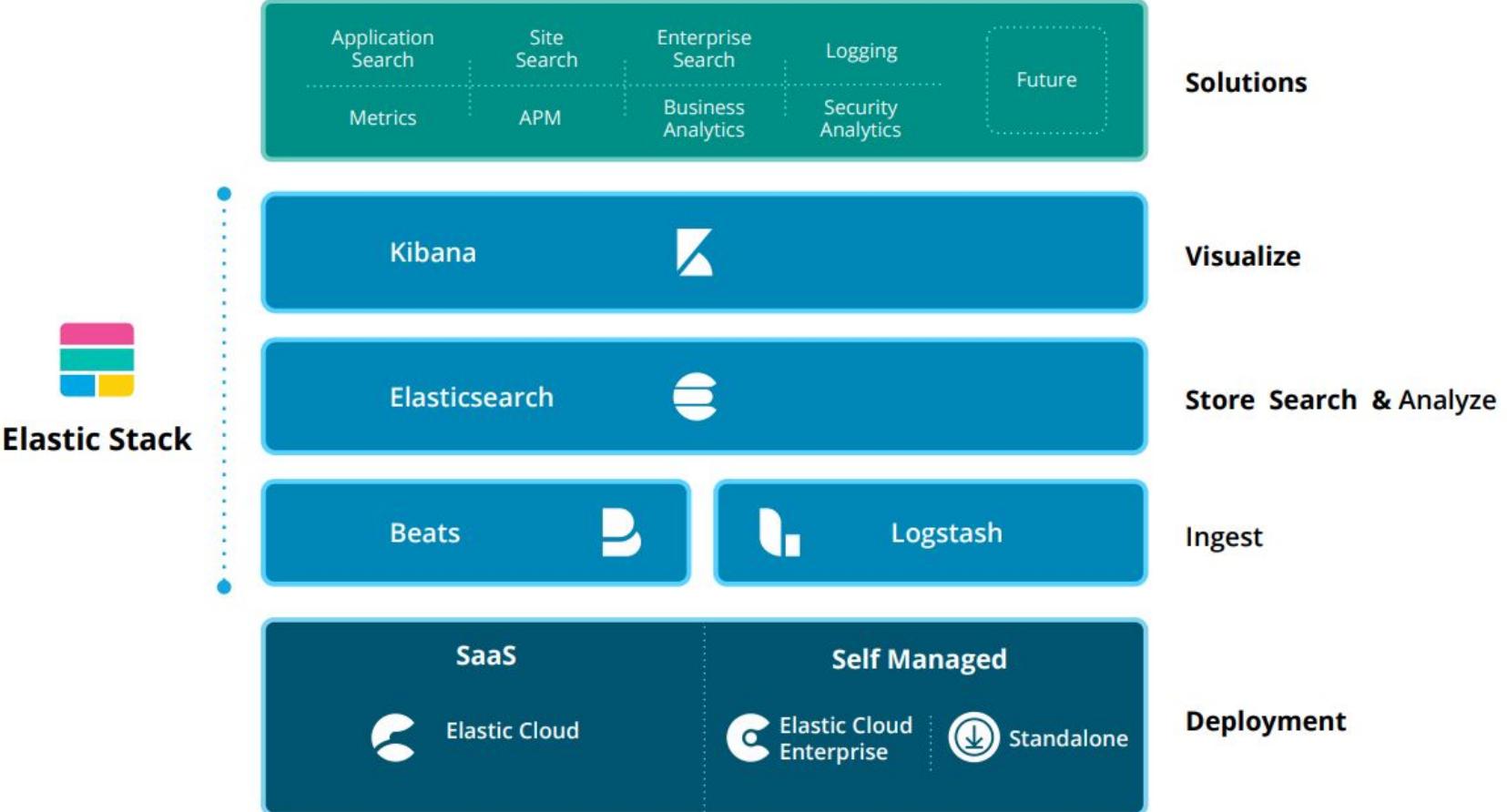
- Can only filter by **exact values**
- Unable to perform **full-text search**
- Queries can be **complex** and **inefficient**
- Often requires **big-batch** processing

Apache Lucene

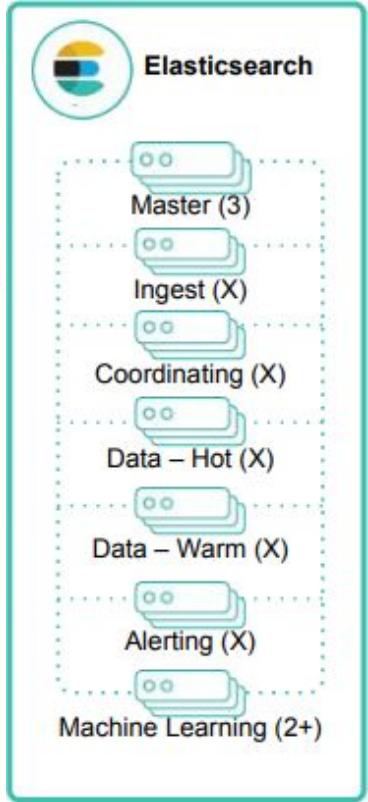
- Arguably, the **best** search engine
- High performance
- Near real-time indexing
- Open source
- ... But
 - It is Java library
 - Hard to use

Elasticsearch



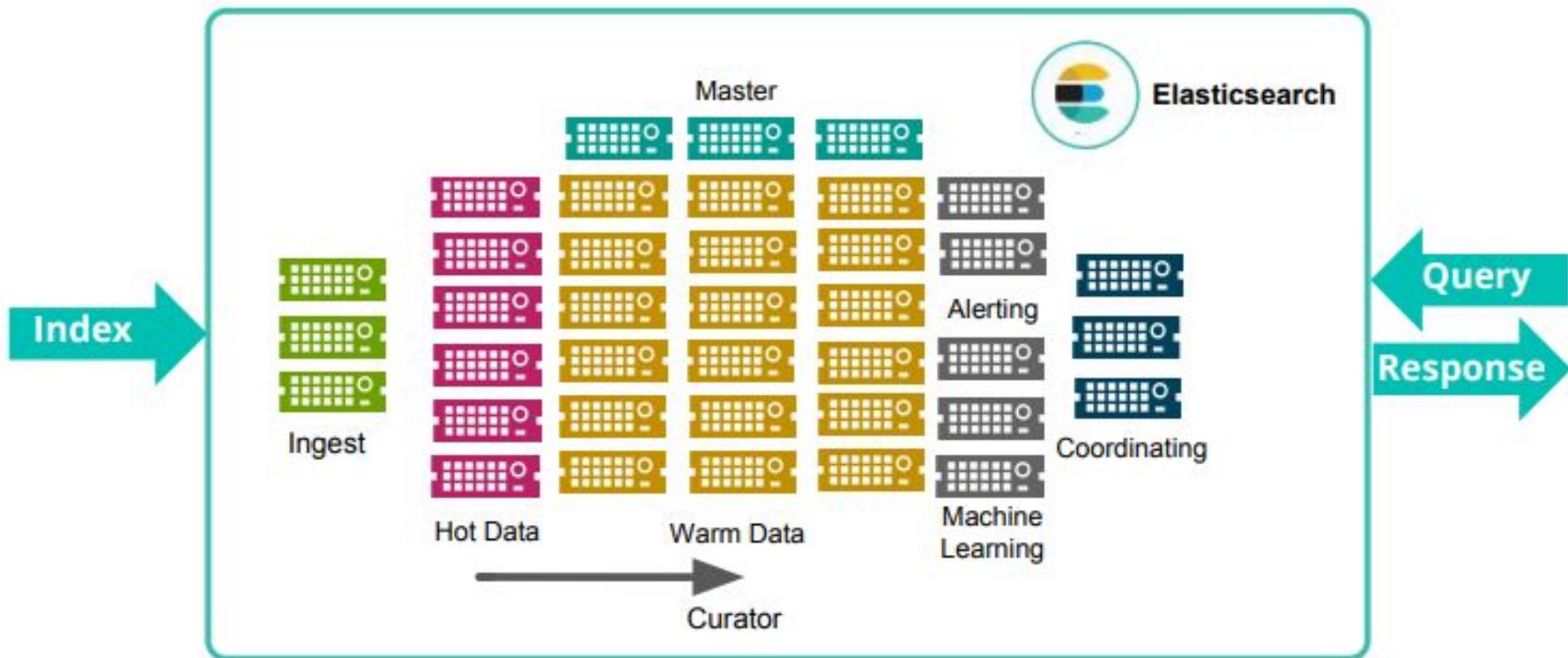


Elasticsearch Node Types



- Master Nodes
 - Control the cluster, requires a minimum of 3, one is active at any given time
- Data Nodes
 - Hold indexed data and perform data related operations
 - Differentiated Hot and Warm data nodes can be used
- Ingest Nodes
 - Use ingest pipelines to transform and enrich before indexing
- Coordinating Nodes
 - Routes request, handle search reduce phase, distribute bulk indexing
 - All nodes function as coordinating nodes
- Alerting Nodes
 - Run alerting jobs
- Machine Learning Nodes
 - Run machine learning jobs

Large cluster of Elasticsearch



Workshop: ELK II

“ Basic command with Elasticsearch “

Prepare system for workshop

- Install Postman from <https://www.getpostman.com/apps>
- Import workshop collection file from <https://bit.ly/2I56SaZ>
- Setup Environment

The Problem with relational databases

- Stores data in **columns** and **rows**
- Equivalent of using a **spreadsheet**
- **Inflexible** storage medium
- Not suitable for **rich objects**

Documents

```
{  
  "name": "John Smith",  
  "age": 42,  
  "confirmed": true,  
  "join_date": "2015-06-01",  
  "home": {"lat": 51.5, "lon": 0.1},  
  "accounts": [  
    {"type": "facebook", "id": "johnsmith"},  
    {"type": "twitter", "id": "johnsmith"}  
  ]  
}
```

Documents metadata

- **Index** - Where the document lives
- **Type** - Class of object that the document represents
- **Id** - Unique identifier for the document

Documents metadata

<i>Relational DB</i>	<i>Databases</i>	<i>Tables</i>	<i>Rows</i>	<i>Columns</i>
<i>Elasticsearch</i>	<i>Indices</i>	<i>Types</i>	<i>Documents</i>	<i>Fields</i>

RESTful API

[VERB] /{index}/{type}/{id}?pretty

GET | POST | PUT | DELETE | HEAD

- **JSON-only**
- Adding **pretty** to the query-string parameters pretty-prints response

Indexing a document with your own ID

PUT /{index}/{type}/{id}

follow request 001 at postman

Response

```
{  
    "_index" : "blog",  
    "_type" : "post",  
    "_id" : "123",  
    "_version" : 1,  
    "_shards" : {  
        "total" : 2,  
        "successful" : 1,  
        "failed" : 0  
    },  
    "created" : true  
}
```

Indexing a document with autogenerated ID

POST /**{index}**/b**{type}**

* Autogenerated IDs are *Base64*-encoded *UUIDs*

Follow request 002 at Postman

Response

```
{  
    "_index" : "blog",  
    "_type" : "post",  
    "_id" : "AVFWIbMf7YZ6Se7RwMws",  
    "_version" : 1,  
    "_shards" : {  
        "total" : 2,  
        "successful" : 1,  
        "failed" : 0  
    },  
    "created" : true  
}
```

Retrieving a document with metadata

GET /{index}/{type}/{id}

Follow request 003 at Postman

Response

```
{  
    "_index": "blog",  
    "_type": "post",  
    "_id": "123",  
    "_version": 1,  
    "found": true,  
    "_source": {  
        "title": "My first blog entry",  
        "text": "Just trying this out...",  
        "date": "2014-01-01"  
    }  
}
```

Retrieving a document without metadata

GET /{index}/{type}/{id}

Follow request 004 at Postman

Response

```
{  
  "title": "My first blog entry",  
  "text": "Just trying this out...",  
  "date": "2014-01-01"  
}
```

Retrieving part of a document

```
GET /{index}/{type}/{id}  
?_source={fields}
```

Follow request 005 at Postman

Response

```
{  
    "_index": "blog",  
    "_type": "post",  
    "_id": "123",  
    "_version": 1,  
    "found": true,  
    "_source": {  
        "title": "My first blog entry",  
        "date": "2014-01-01"  
    }  
}
```

Checking whether a document exists

HEAD /{index}/{type}/{id}

Follow request 006 at Postman

Response

HTTP/1.1 200 OK

Content-Length: 0

Response

HTTP/1.1 404 Not Found

Content-Length: 0

Updating a whole document

PUT /{index}/{type}/{id}

Follow request 008 at Postman

Response

```
{  
    "_index" : "blog",  
    "_type" : "post",  
    "_id" : "123",  
    "_version" : 2,  
    "_shards" : {  
        "total" : 2,  
        "successful" : 1,  
        "failed" : 0  
    },  
    "created" : false  
}
```

Deleting a document

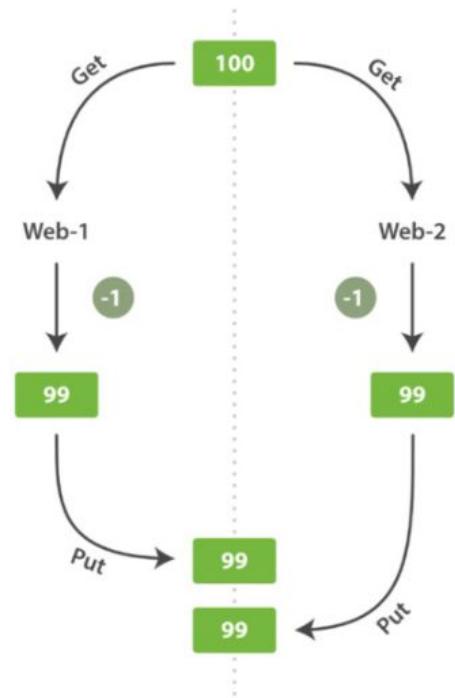
DELETE /{index}/{type}/{id}

Follow request 009 at Postman

Response

```
{  
    "found" : true,  
    "_index" : "blog",  
    "_type" : "post",  
    "_id" : "123",  
    "_version" : 3,  
    "_shards" : {  
        "total" : 2,  
        "successful" : 1,  
        "failed" : 0  
    }  
}
```

Dealing with conflict



Pessimistic concurrency control

- Used by relational databases
- Assumes conflicts are likely to happen (**pessimist**)
- **Blocks** access to resources

Optimistic concurrency control

- Assumes conflicts are unlikely to happen (**optimist**)
- Does **not** block operations
- If conflict happens, update **fails**

How Elasticsearch deals with conflicts

- Locking distributed resources would be very **inefficient**
- Uses **Optimistic Concurrency Control**
- Auto-increments **_version** number

How Elasticsearch deals with conflicts

- **PUT /blog/post/123?version=1**
- If version is outdated returns **409 Conflict**

Search Example

Employee directory example

- Index: megacorp
- Type: employee
- Example records: John Smith, Jane Smith, Douglas Fir

Follow request 010 - 012 at Postman

Searches all employees

GET /megacorp/employee/_search

Follow request 013 at Postman

Search with Query-String

```
GET /megacorp/employee/_search  
?q=last_name:Smith
```

Follow request 014 at Postman

Response

```
"hits" : {  
    "total" : 2,  
    "max_score" : 0.30685282,  
    "hits" : [ {  
        ...  
        "_score" : 0.30685282,  
        "_source": {  
            "first_name": "Jane",  
            "last_name": "Smith", ... }  
    }, {  
        ...  
        "_score" : 0.30685282,  
        "_source": {  
            "first_name": "John",  
            "last_name": "Smith", ... }  
    } ]  
}
```

Search with Query DSL

```
{  
  "query": {  
    "match": {  
      "last_name": "Smith"  
    }  
  }  
}
```

Follow request 015 at Postman

Response

```
"hits" : {  
    "total" : 2,  
    "max_score" : 0.30685282,  
    "hits" : [ {  
        ...  
        "_score" : 0.30685282,  
        "_source": {  
            "first_name": "Jane",  
            "last_name": "Smith", ... }  
    }, {  
        ...  
        "_score" : 0.30685282,  
        "_source": {  
            "first_name": "John",  
            "last_name": "Smith", ... }  
    } ]  
}
```

Search with Query DSL and filtered

```
{  
  "query": {  
    "bool": {  
      "must": {  
        "match": { "last_name": "Smith" }  
      },  
      "filter": {  
        "range": {  
          "age": { "gt": 30 }  
        }  
      }  
    }  
  }  
}
```

Follow request 016 at Postman

Response

```
"hits" : {  
    "total" : 1,  
    "max_score" : 0.30685282,  
    "hits" : [ {  
        ...  
        "_score" : 0.30685282,  
        "_source": {  
            "first_name": "Jane",  
            "last_name": "Smith",  
            "age": 32, ... }  
    } ]
```

Full-text search

```
{  
  "query": {  
    "match": {  
      "about": "rock climbing"  
    }  
  }  
}
```

Follow request 017 at Postman

Response

```
"hits" : [{ ...  
    "_score" : 0.16273327,  
    "_source": {  
        "first_name": "John", "last_name": "Smith",  
        "about": "I love to go rock climbing", ... }  
    }, { ...  
        "_score" : 0.016878016,  
        "_source": {  
            "first_name": "Jane", "last_name": "Smith",  
            "about": "I like to collect rock albums", ... }  
    }]
```

Relevance Scores

- The **_scores** field rank search results
- The **higher** the score, the **better**

Phrase search

```
{  
  "query": {  
    "match_phrase": {  
      "about": "rock climbing"  
    }  
  }  
}
```

Follow request 018 at Postman

Response

```
"hits" : {  
    "total" : 1,  
    "max_score" : 0.23013961,  
    "hits" : [ {  
        ...  
        "_score" : 0.23013961,  
        "_source": {  
            "first_name": "John",  
            "last_name": "Smith",  
            "about": "I love to go rock climbing"  
            ... }  
        } ]  
}
```

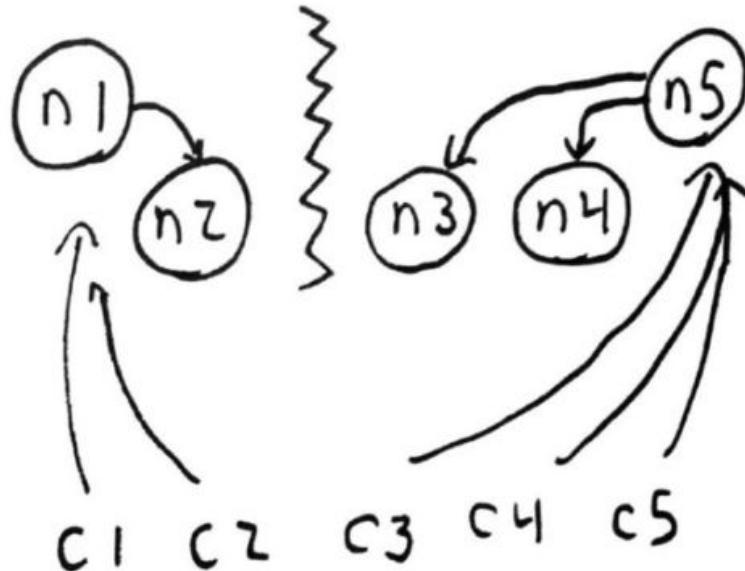
Data Resiliency

Call Me Maybe

- **Jepsen Tests**
- Simulates **network partition** scenarios
- Run several operations against a distributed system
- Verify that the history of those operations makes sense

Elasticsearch Status

- Risk of data loss on network partition and **split-brain** scenarios



It is not so bad...

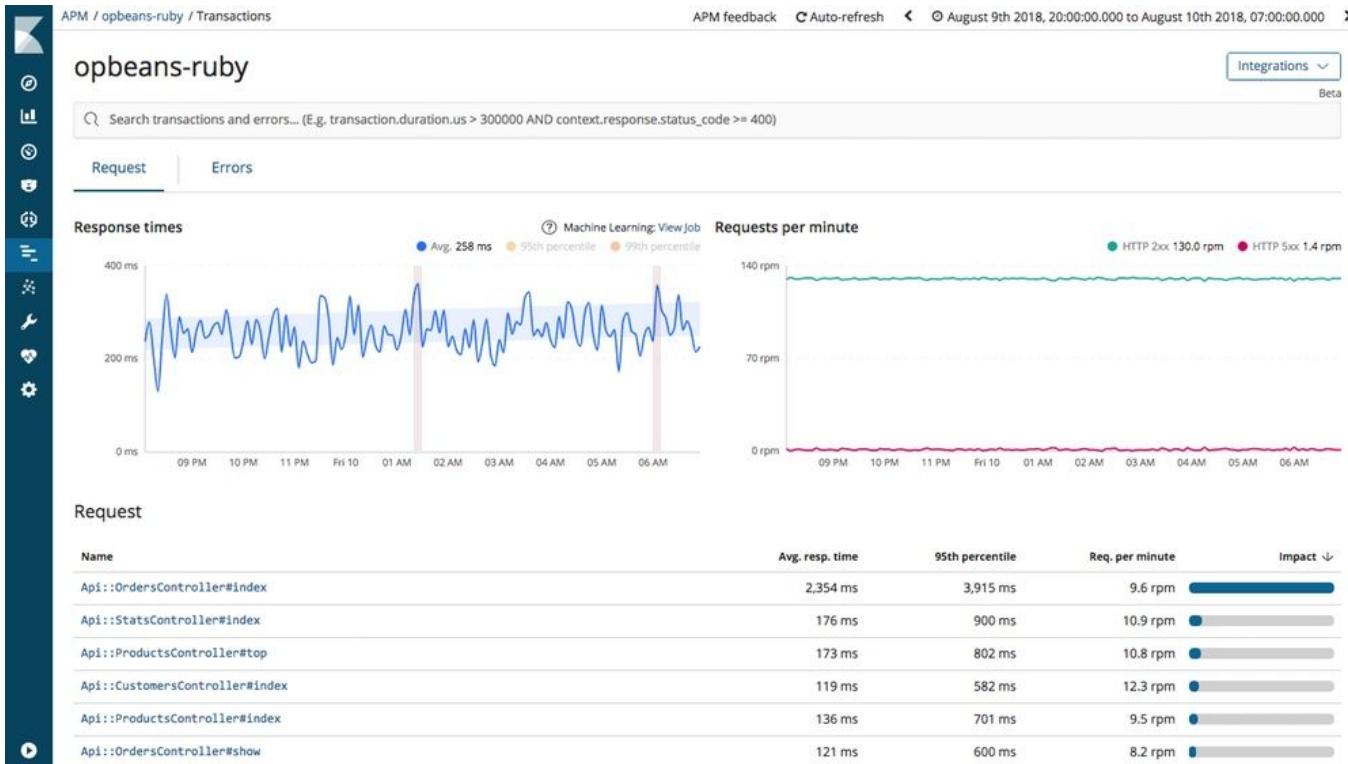
- Still much more resilient than **MongoDB**
- Elastic Co is working hard to improve it
- **Two-phase commits** are planned

If you really care about your data

- Use a more reliable **primary** data store:
 - Cassandra
 - Postgres
 - etc.
- **Synchronize** it to Elasticsearch
- ... or set-up comprehensive **back-up**

*There's **no such thing** as a 100%
reliable distributed system*

Application Performance Monitoring



Application Performance Monitoring

APM Server

APM Client

Setup APM Server

docker.elastic.co/apm/apm-server:6.3.1

port : 8200 (shouldn't public)

Select Image*

Always pull image before creating

docker.elastic.co/apm/apm-server:6.3.1

[Port Map](#)

[Service Links](#)

Destination Service

As Name

es-cluster/es-client > elasticsearch

-

Setup APM client

1 Install the APM agent

Install the APM agent for Node.js as a dependency to your application.

[Copy snippet](#)

```
npm install elastic-apm-node --save
```

2 Configure the agent

Agents are libraries that run inside of your application process. APM services are created programmatically based on the `serviceName`. This agent supports Express, Koa, hapi, and custom Node.js.

[Copy snippet](#)

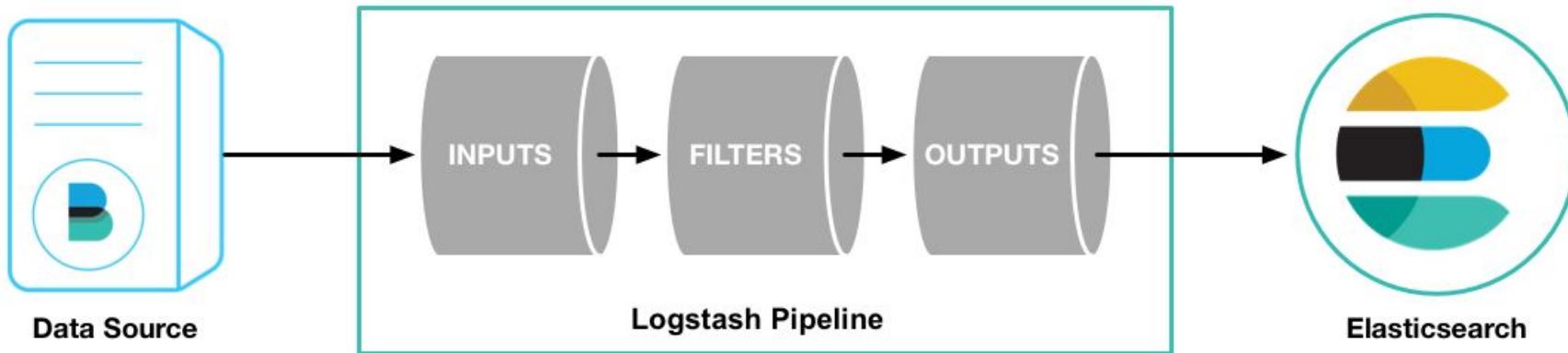
```
// Add this to the VERY top of the first file loaded in your application
var apm = require('elastic-apm-node').start({
  // Set required service name (allowed characters: a-z, A-Z, 0-9, -, _, and space)
  serviceName: '',
  // Use if APM Server requires a token
  secretToken: '',
  // Set custom APM Server URL (default: http://localhost:8200)
  serverUrl: ''
})
```

Workshop: APM

“ Setup npm server and npm client with nodejs ”

Lunch

Logstash



Logstash: Inputs

```
udp {  
    port => 5000  
    codec => "plain"  
}
```

From load balance -> Logstash

SSL Termination

Stickiness

Custom haproxy.cfg

Labels

Scheduling

Additional [haproxy.cfg](#) configuration can be put here and will be merged together with the configuration generated by Rancher.

```
log 10.42.141.227:5000 local0
log global
option redispatch

frontend 80
option httplog
option http-buffer-request
# declare a capture slot giving it a max length (400) and a reference (0)
declare capture request len 4000
capture request header Host len 100
capture request header Via len 100
capture request header yguid len 52
capture request header X-Forwarded-For len 50
capture request header X-Access-Token len 300
http-request capture req.body id 0
```

From load balance -> Logstash

```
global
maxconn 2048

defaults
log 127.0.0.1:8514 local0 # Log to Rancher
log 10.42.141.227:5000 local0 # Change ip here
log global
option redispatch

frontend 80
option httplog
option http-buffer-request
declare capture request len 4000
capture request header Host len 100
capture request header Via len 100
capture request header yguid len 52
capture request header X-Forwarded-For len 50
capture request header X-Access-Token len 300
http-request capture req.body id 0
#http-request capture req.payload(0,1000) len 1000
http-request add-header X_FORWARDED_FOR %[req.hdr(CF-Connecting-IP)]
#log-format {"%t|Z=%Ts|code=%ST|m=%HM|H=%[capture.req.hdr(1)]|AB=%cp|A=%[capt
log-format "%t %Ts %ST %HM %[capture.req.hdr(1)] %cp %[capture.req.hdr(4)] %[
|
```

Logstash: Filters

```
grok {  
    |   match => { "message" => "%{NOTSPACE:time} %{INT:timestamp} %{INT:status_code} %{WOR  
}  
}  
date {  
    |   match => [ "time", "dd/MMM/yyyy:HH:mm:ss.SSS" ]  
}  
  
if "_grokparsefailure" in [tags] { drop {} }  
}
```

Logstash: Grok

Grok Debugger

Sample Data

```
1 |
```

Grok Pattern

```
1 |
```

▶ Custom Patterns

Simulate

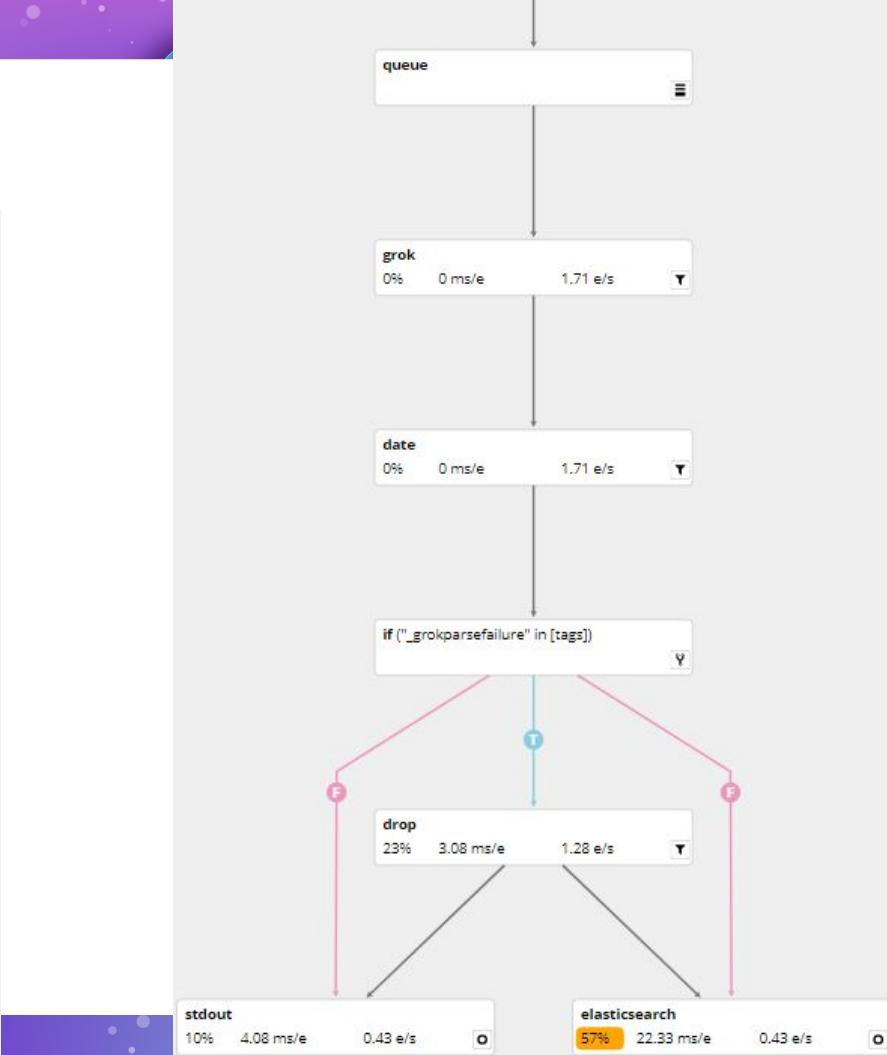
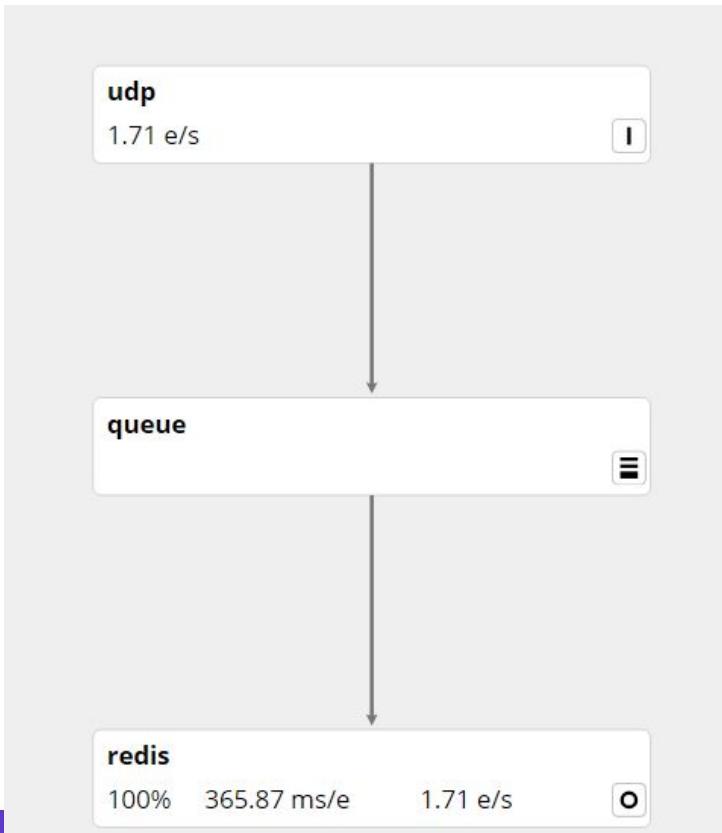
Structured Data

```
1 | { }
```

Logstash: Outputs

```
elasticsearch {  
    hosts => ["elasticsearch.rancher.internal:9200"]  
}  
stdout {  
    codec => rubydebug  
}
```

Handle bottleneck



Setup Logstash

 logstash

Category: All ▾



Manage 



Logstash

Centralize data processing of all types

[View Details](#)

Setup Logstash

New Stack

Name*

logstash

Description

Description

Configuration Options

Logstash inputs*

```
udp {  
    port => 5000  
    codec => "plain"  
}
```

```
grok {  
    match => { "message" => "%{NOTSPACE:time} %{INT:timestamp} %{INT:status_code} %{WORD:method} %{NOTSPACE:host} %{NOTSPACE:client_port} %{IP:client_ip} %{NOTSPACE:Via} %{NOTSPACE:yguid} %{NOTSPACE:query} %{NOTSPACE:byte} %{NOTSPACE:bknd} %{NOTSPACE:bkndq} %{NOTSPACE:srvq} %{NOTSPACE:actconn} %{NOTSPACE:feconn} %{NOTSPACE:beconn} %{NOTSPACE:srvconn} %{NOTSPACE:retr} %{NOTSPACE:Tq} %{NOTSPACE:Tw} %{NOTSPACE:Tc} %{NOTSPACE:Tr} %{NOTSPACE:Tt} %{NOTSPACE:Authorization} %{GREEDYDATA:data}" }  
}  
date {  
}
```

Logstash collection tier inputs. These will be added directly to input [] section of logstash.conf



Logstash outputs*

```
elasticsearch {  
    hosts => ["elasticsearch.rancher.internal:9200"]  
}  
stdout {  
    codec => rubydebug  
}
```

Elasticsearch stack/service*

es-cluster/es-client



stack/service link or external service link to elasticsearch cluster.

Demo: Centralized Logging

“ Setup Centralized Logging “

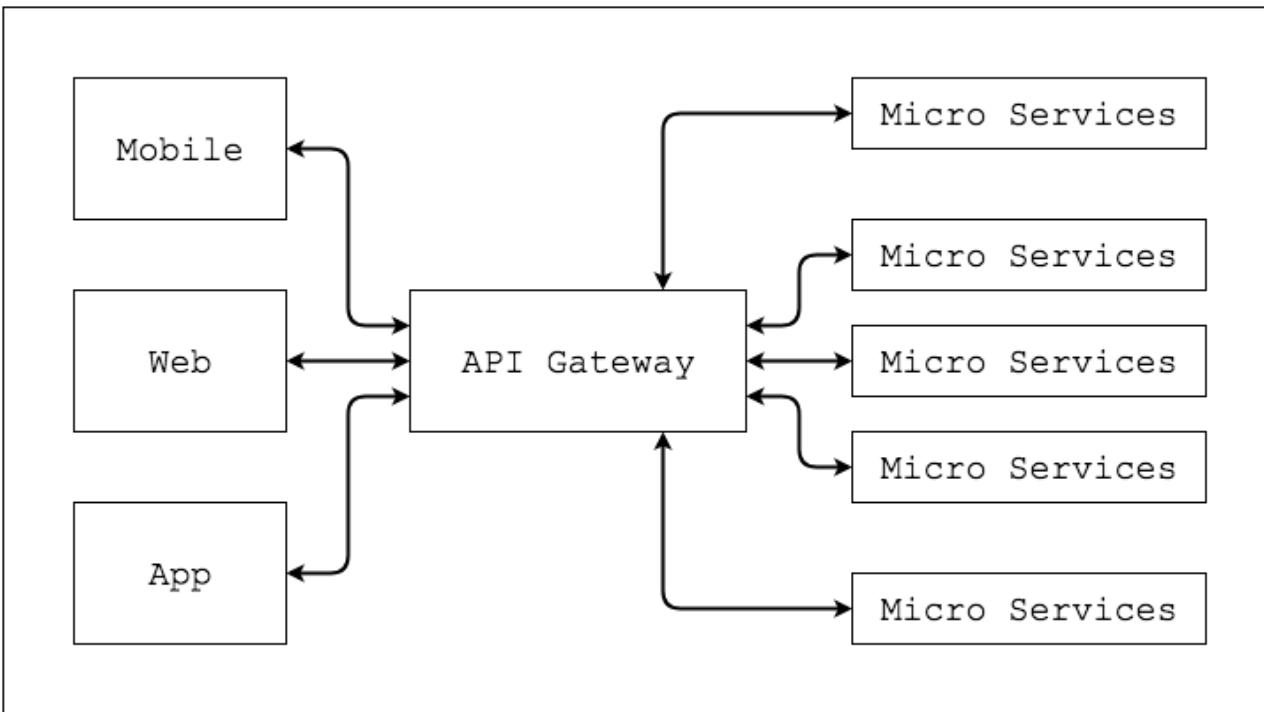
Workshop: Centralized Logging

“Centralized Logging with Haproxy and Logstash”

Break

Kong API Gateway

Kong



Why API Gateway ?

ลดการทำงานเดิมซ้ำๆ
และควบคุมการใช้งาน API ได้จากที่เดียว

Why Kong ?



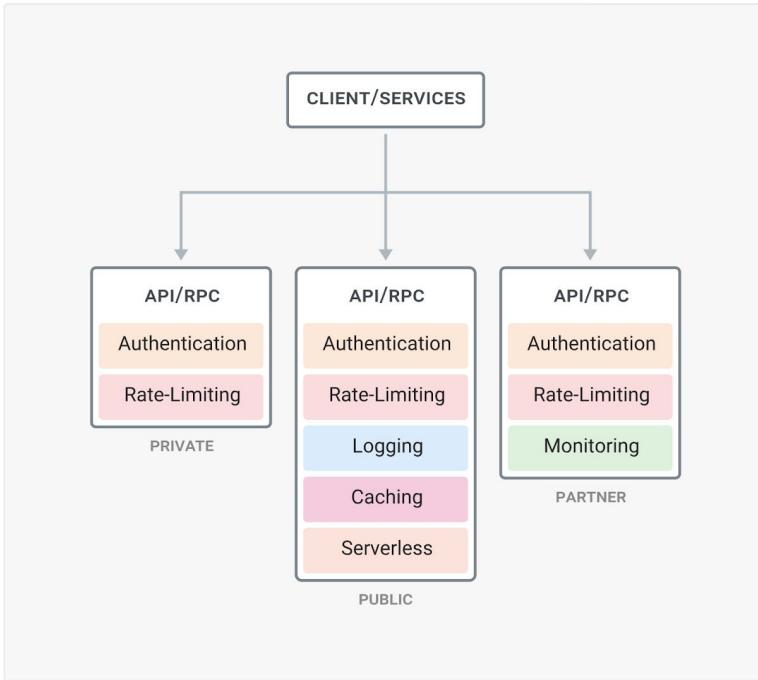
Why Kong ?

- OpenSource
- Scale
- Fast
- **RESTful**
- Plugins
- **NGINX + Lua**

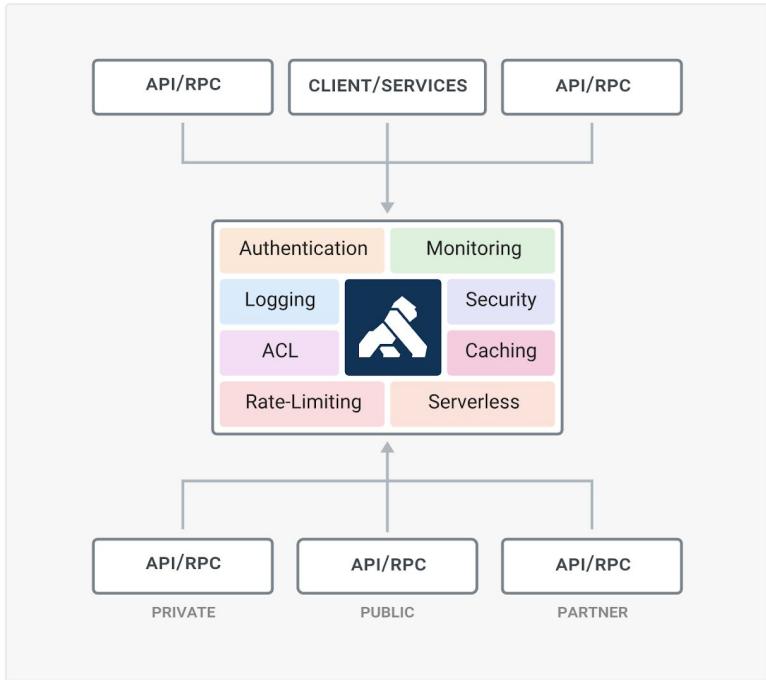


Kong

The Redundant Old Way



The Kong Way



NGINX

NGINX อ่านว่า เอ็นจีนเอ็กซ์
“ไม่ใช่ หิง”
มาจาก Engine-X

NGINX

NGINX คือ Web Server ที่ทำได้มากกว่าเป็นแค่ Web Server

- Reverse Proxy Server
- Cache Server
- Load Balancer



NGINX + Lua = OpenResty

สามารถเขียนโปรแกรมด้วยภาษา Lua (LuaJIT) ไปรันใน NGINX ได้



ก่อนใช้ KONG

NGINX สามารถใช้เป็น API Gateway ได้ โดยการทำ Reverse Proxy

NGINX Reverse Proxy

```
server {
# skip server config
location /v1/ {
    proxy_pass http://127.0.0.1:3000/;
}
location /v2/ {
    proxy_pass http://127.0.0.1:3001/;
}
}
```

NGINX Reverse Proxy

nginx -t

nginx -s reload

เอະอะກີ ຄອມມານດໍໄລນ໌

Demo: Nginx

ปัญหา ของการทำ API Gateway ด้วย NGINX

1. ต้องใช้ NGINX เป็น, *static config*
2. ต้อง Reload config
3. ต้องมีสิทธิ์เข้าถึง Server (ssh)

KONG API Gateway



Kong

KONG Services

Service คือ
Object ของ API ของเรา

KONG Routes

Route คือ
การกำหนดให้ Request ส่งไปยัง Service

KONG Plugins

KONG สามารถเพิ่ม Plugins ได้ เช่น

- Basic Authentication
- Key Authentication
- Rate Limit
- Bot Detection
- Custom Plugin (Lua Programming)
- etc.

KONG ADMIN and PROXY

ADMIN ใช้สำหรับคอนฟิกค่าต่างๆ
PROXY สำหรับ Request ที่จะส่งไปยัง Service

ตัวอย่าง: สร้าง KONG Service

```
curl -s 127.0.0.1:8001/services \
-d 'name=api-v1' \
-d 'url=http://127.0.0.1:3000'
```

ตัวอย่าง: สร้าง KONG Routes

```
curl -s 127.0.0.1:8001/routes \
-d 'hosts[]='devops-bkk.dont.works' \
-d 'service.id=<SERVICE_ID>' \
-d 'paths[]=/v1'
```

Test with cURL

```
curl devops-bkk.dont.works/v1/posts/1
{
  "id": 1,
  "title": "json-server",
  "author": "typicode"
}
```

เพิ่ม KONG Plugins ให้กับ Service

1. เพิ่ม Plugins Key Authentication ให้กับ API
2. เพิ่ม Plugins
3. สร้าง Consumer
4. สร้าง Key สำหรับ Consumer
5. ทดสอบ

เพิ่ม Plugins key-auth

```
curl 127.0.0.1:8001/services/api-v1/plugins \
-d 'name=key-auth'
{
  "created_at": 1536146037608,
  "config": { "key_names": [ "apikey" ] },
  "id": "f51b8d8b-e23f-4951-acd2-f3f869489450",
  "service_id": "12fe46d9-fa17-4878-a63b-236b5d482bed",
  "name": "key-auth", "enabled": true, ...
}
```

สร้าง Consumers

```
curl 127.0.0.1:8001/consumers \
-d 'username=kob'
{
  "custom_id": null,
  "created_at": 1536146315,
  "username": "kob",
  "id": "35e9dce9-3e9b-4a75-8a3e-f09542fb6e1e"
}
```

สร้าง Key สำหรับ kob

```
curl 127.0.0.1:8001/consumers/kob/key-auth \
-d ''
{
  "id": "32a9b91e-447d-4dae-9ba9-e6e047e11667",
  "created_at": 1536146399679,
  "key": "rzFTaBXlUGEtAAW03AsCeZ8bac5U0Nc",
  "consumer_id": "35e9dce9-3e9b-4a75-8a3e-f09542fb6e1e"
}
```

Test with cURL

```
curl devops-bkk.dont.works/v1/posts/1
{
  "message": "No API key found in request"
}
```

Test with cURL

```
curl -H 'apiKey: rzFTaBXlUGEtAAW03A...' \
devops-bkk.dont.works/v1/posts/1
{
  "id": 1,
  "title": "json-server",
  "author": "typicode"
}
```

Kong

Demo By P'นเรศ 1.5-2Hr

ເວລະກີ Command line



Install KONG (docker-compose)

<https://github.com/narate/kong-docker>

More than just another GUI to Kong Admin API

<https://github.com/pantsel/konga>

KONGA

DASHBOARD

API GATEWAY

- INFO
- SERVICES
- ROUTES
- APIS (DEPRECATED)
- CONSUMERS
- PLUGINS
- UPSTREAMS
- CERTIFICATES

APPLICATION

- USERS
- CONNECTIONS
- SNAPSHOTS
- SETTINGS

CONNECTIONS

Total Requests: 210

ACTIVE	READING	WRITING	WAITING	ACCEPTED	HANDED
2	0	2	0	209	209

NODE INFO

HostName	e177f0fe826e
Tag Line	Welcome to kong
Version	0.14.1
LUA Version	LuaJIT 2.1.0-beta3
Admin listen	["0.0.0.0:8001"]

TIMERS

Pending

Running

DATASTORE INFO

Reachable

DBMS	cassandra
Contact points	kong-database
Keyspace	kong
Data Centers	dc1:2, dc2:3
Username	kong
Port	9042

PLUGINS

KONGA 0.12.2 GitHub Issues

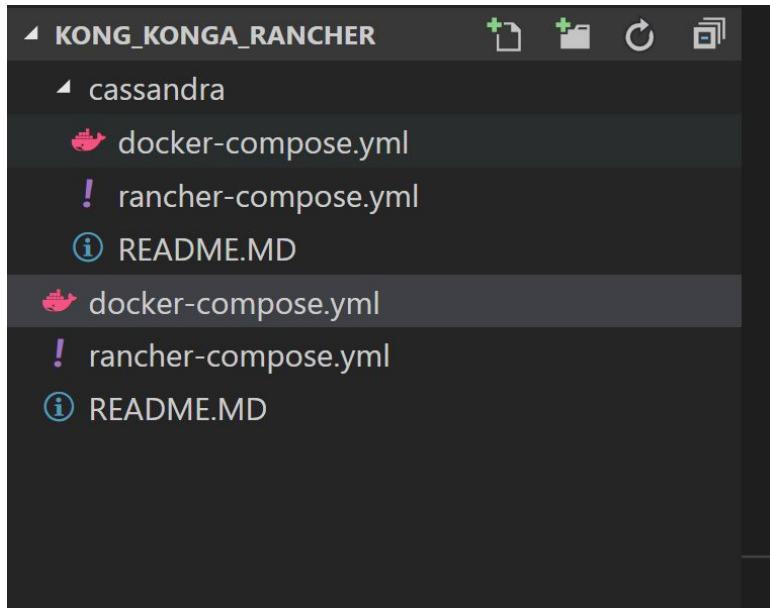
Connected to local

KONG Workshop

- RESTful
- Service
- Routes
- Plugin
- Konga ** GUI **

\$ curl

Setup Kong and Konga



github.com/lma8/kong_konga_rancher

Demo & Workshop: Kong I

“ Setup Kong and Konga “

Workshop: Kong II

“ How to use kong “

Workshop: Kong IV

“ Use kong for example api “

Workshop: Kong V

“ Use kong for kibana “

Summary

- Concept idea
- Docker
- High availability Service/Database
- CI/CD Pipelines
- Measure application
- Investigate with logging
- API Gateway

แมวไม่ว่าจะสีอะไรขอให้จับหนูได้ก็เป็นพ่อ

Thank you.