WISSEN

Exploring Partnership Opportunities

NAGENDRA KUMAR

Nagendra.kommana@wissen.com
(+91) (9989339903)

WISSEN

docker

for beginners

Objectives

- What is Docker?
- What are Containers?
- Why do you need it?
- What can it do?
- Docker for Windows/Mac
- Run Docker Containers
- Create a Docker Image
- Networks in Docker
- Docker Compose
- Docker Swarm
- Docker Concepts in Depth

WISSEN

docker

overview

Docker Inc.

Docker Inc.

dotCloud → Docker

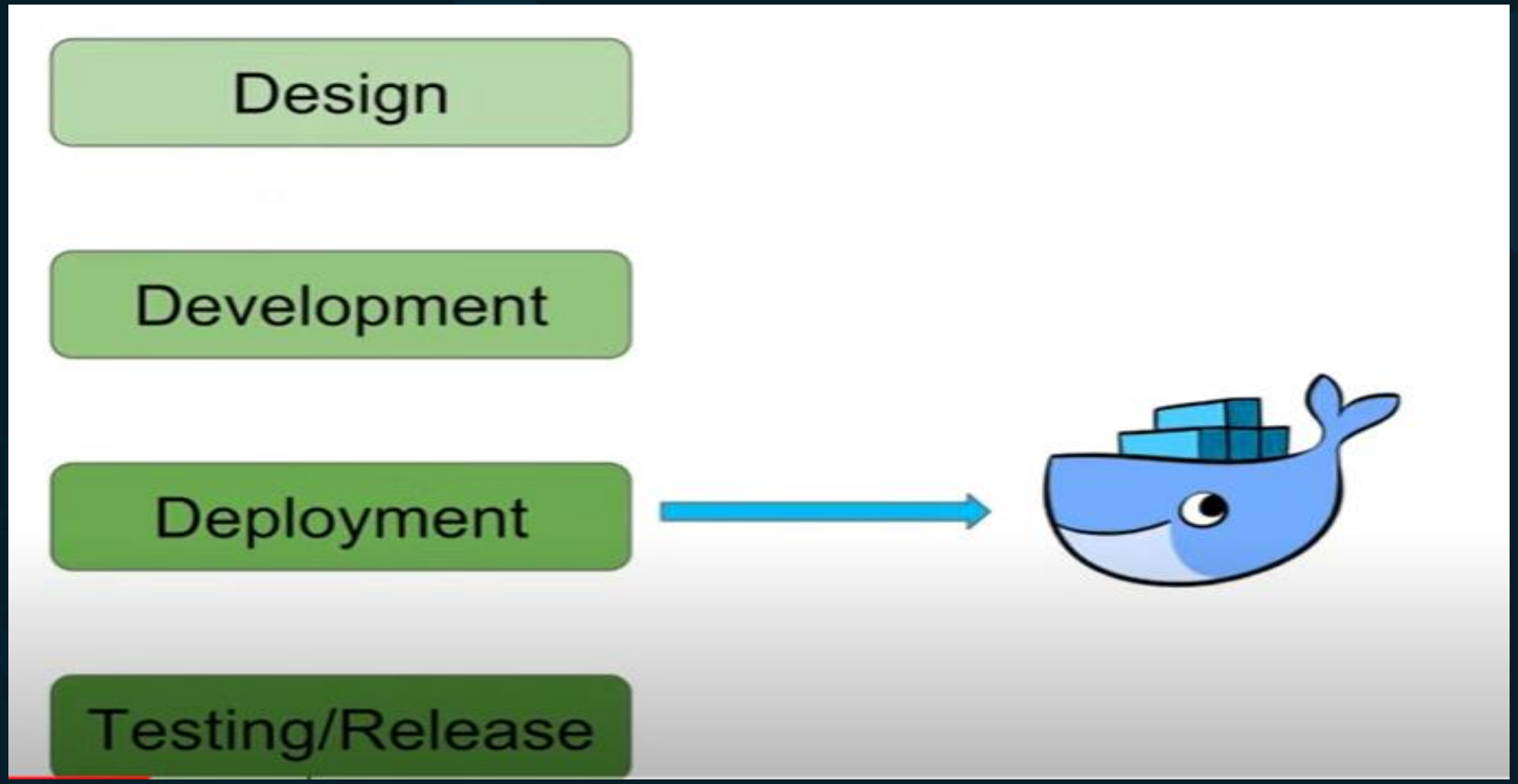
Solomon Hykes

Dock ~~worker~~

$1.5B

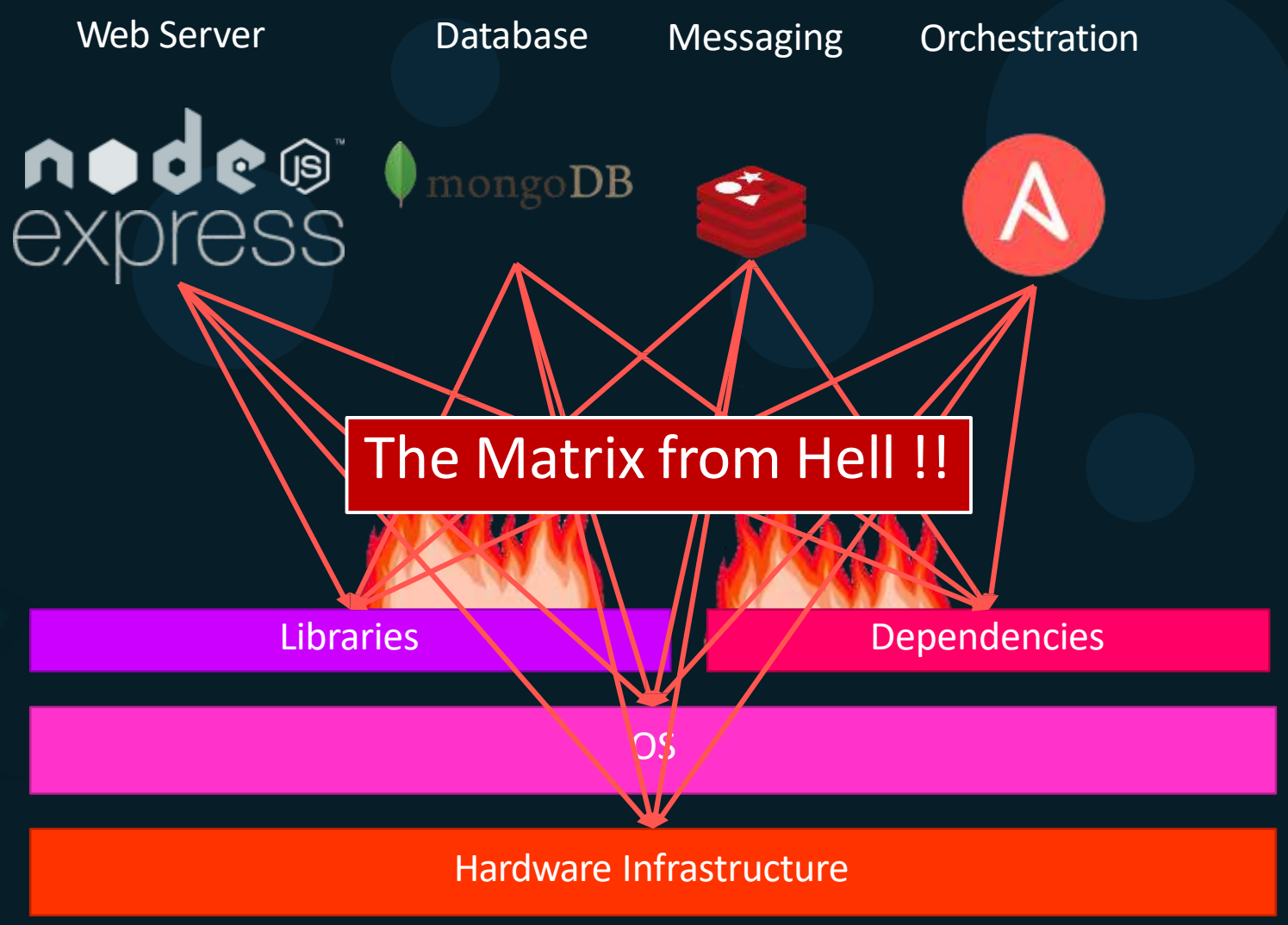
Go

Where Docker comes ?



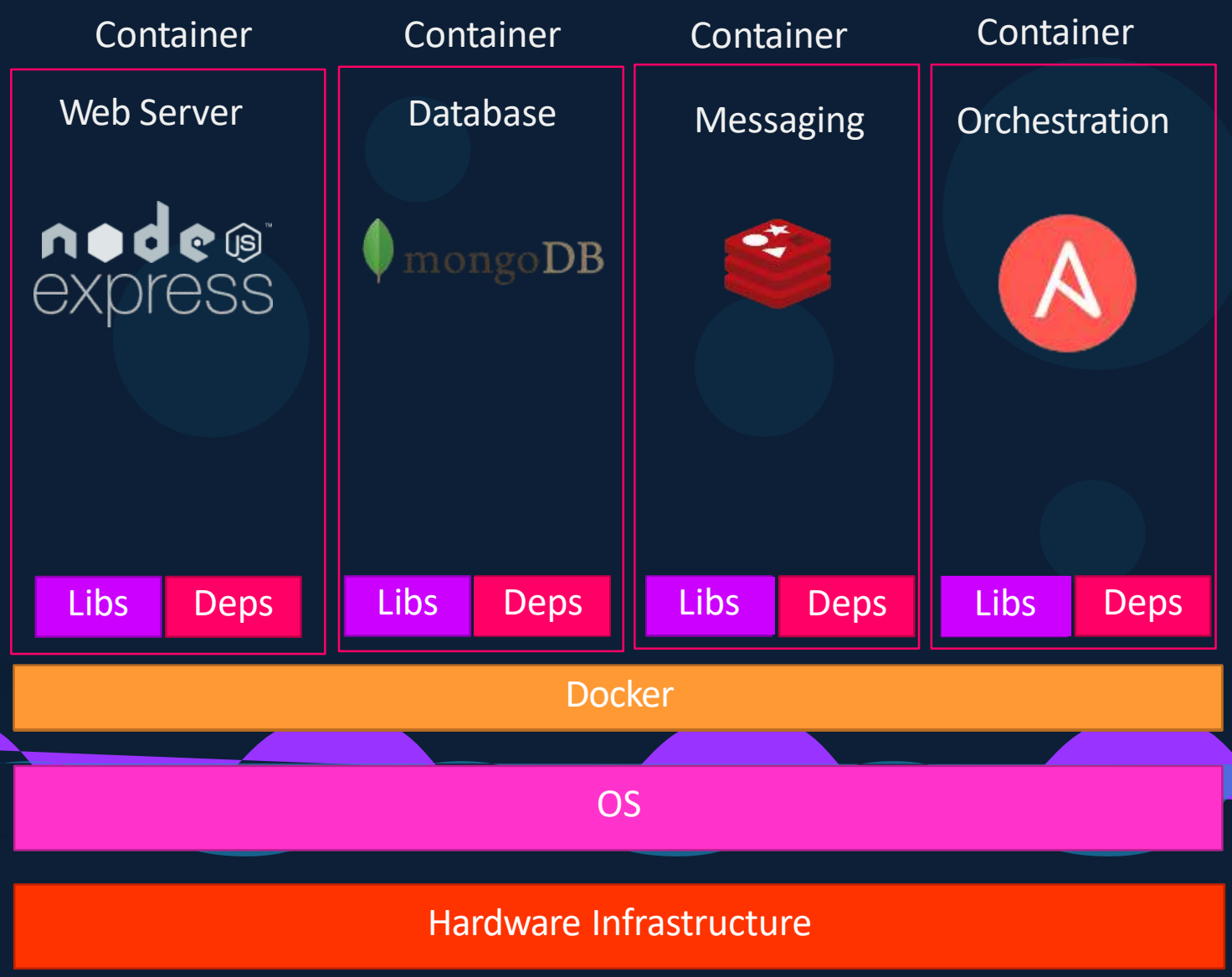
Why do you need docker?

- Compatibility/Dependency
- Long setup time
- Different Dev/Test/Prod environments

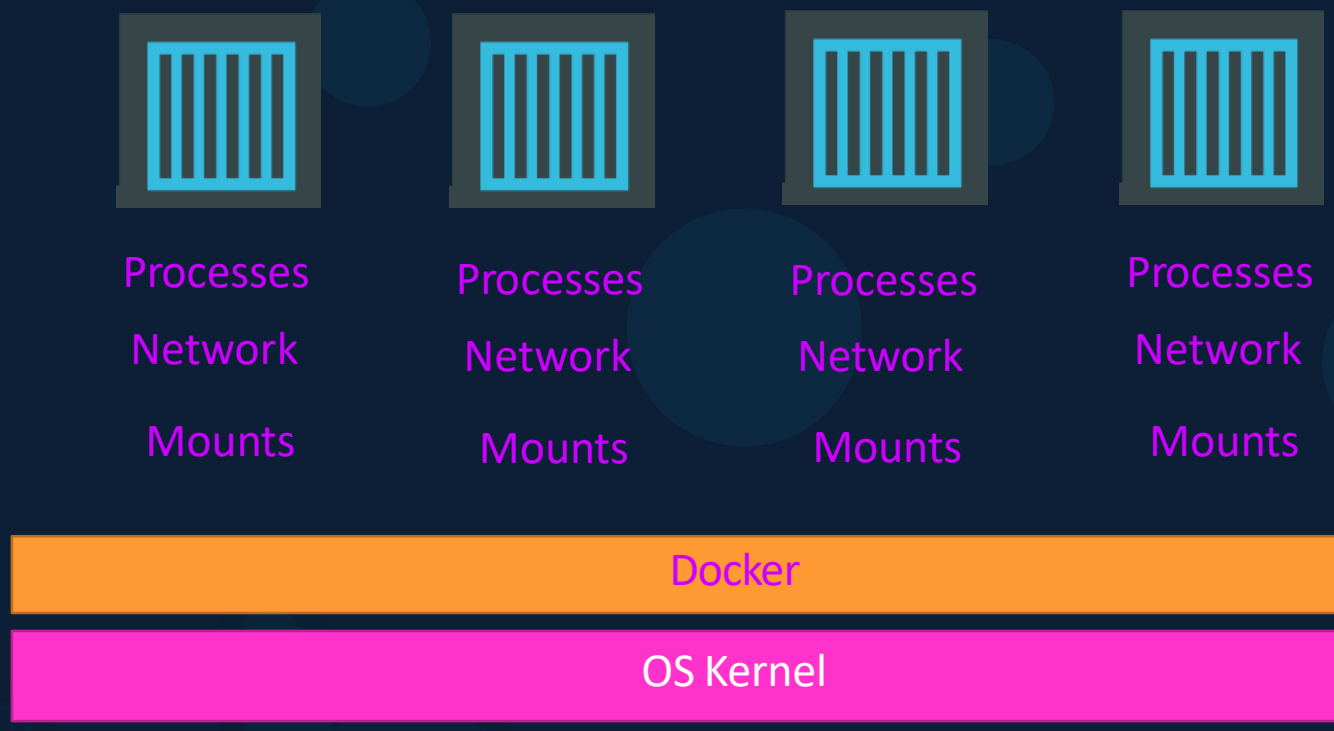


What can it do?

- Containerize Applications
- Run each service with its own dependencies in separate containers

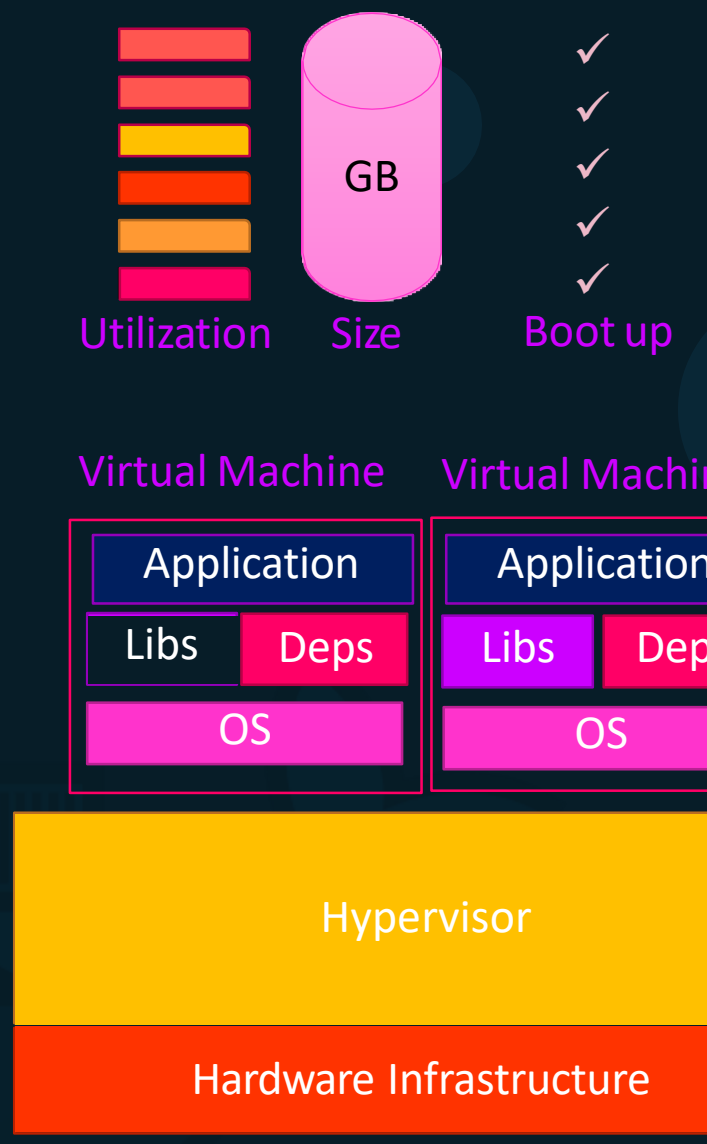


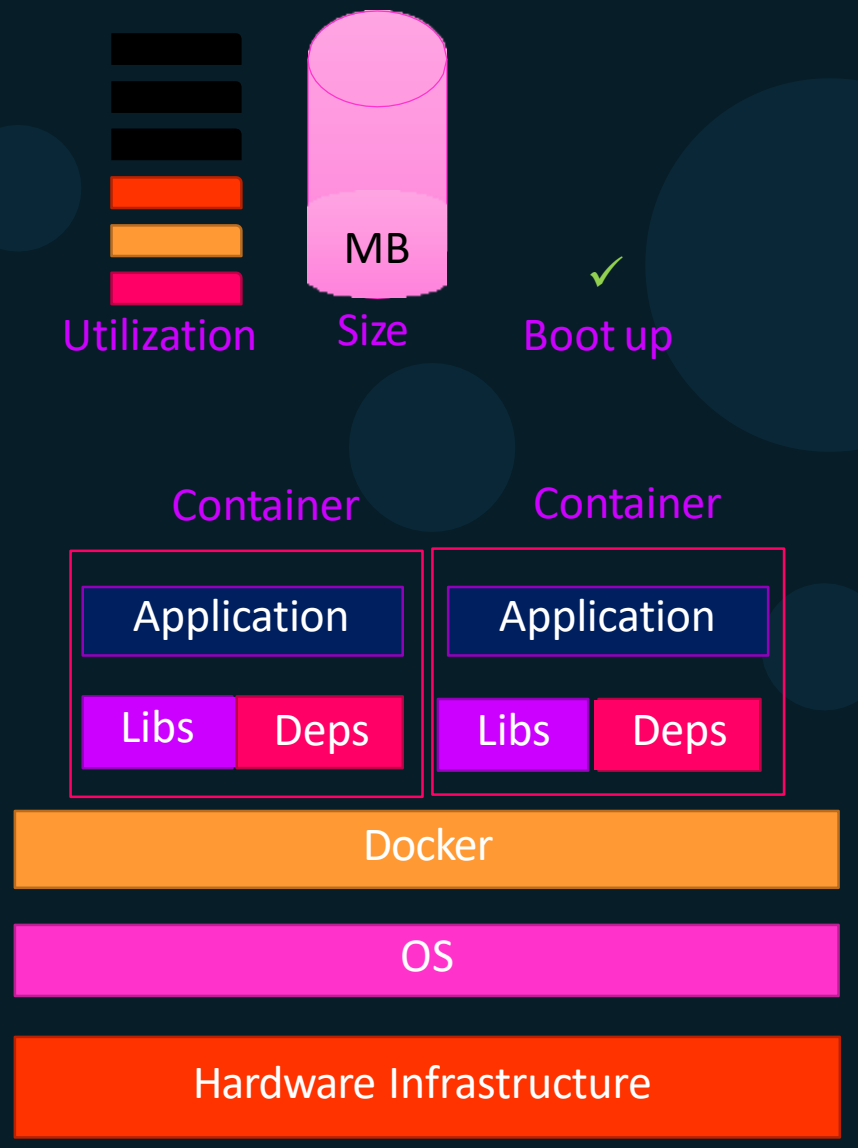
What are containers?



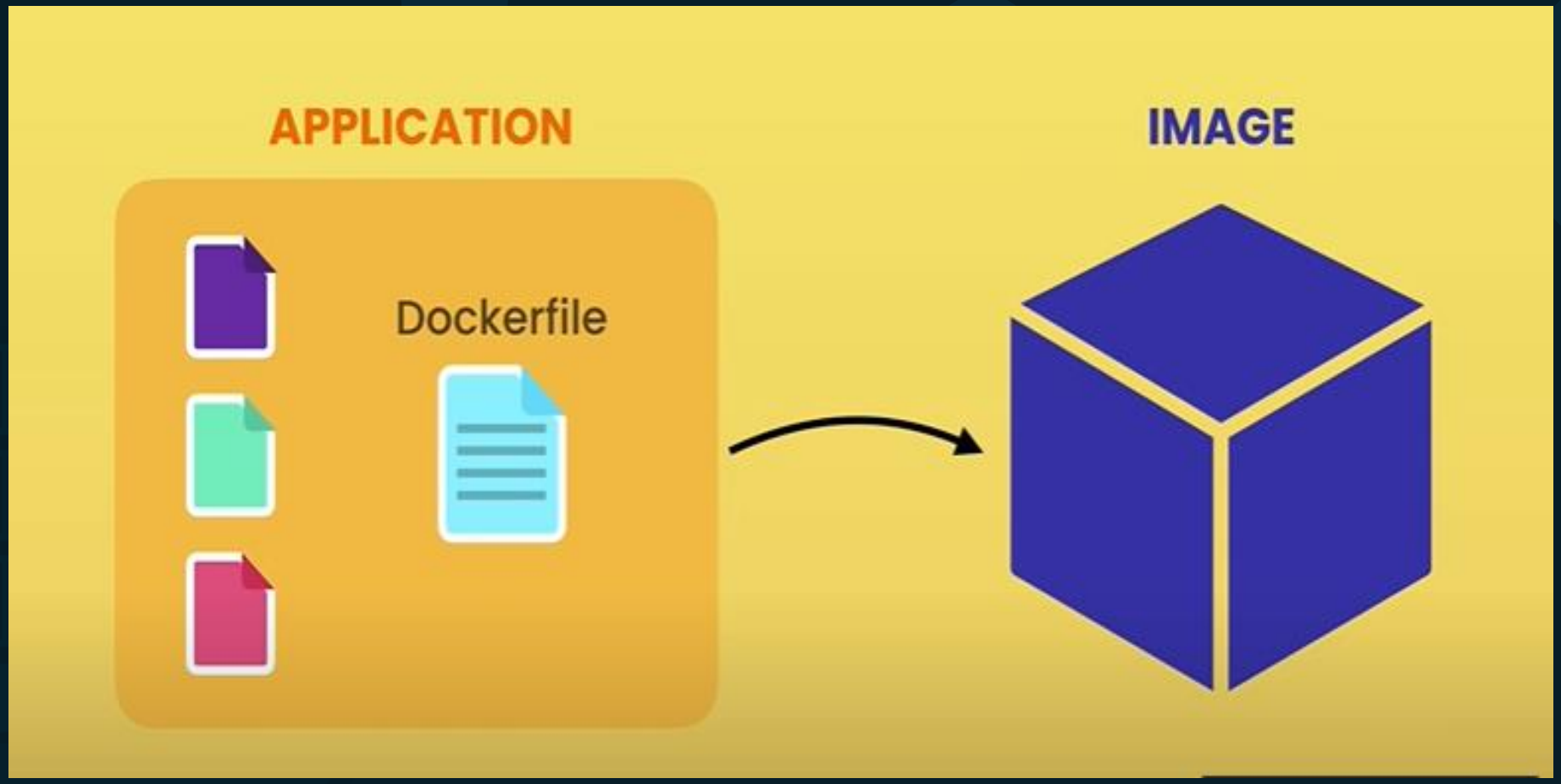
Container

Containers vs Virtual Machines





Image



Container vs image

Docker Image

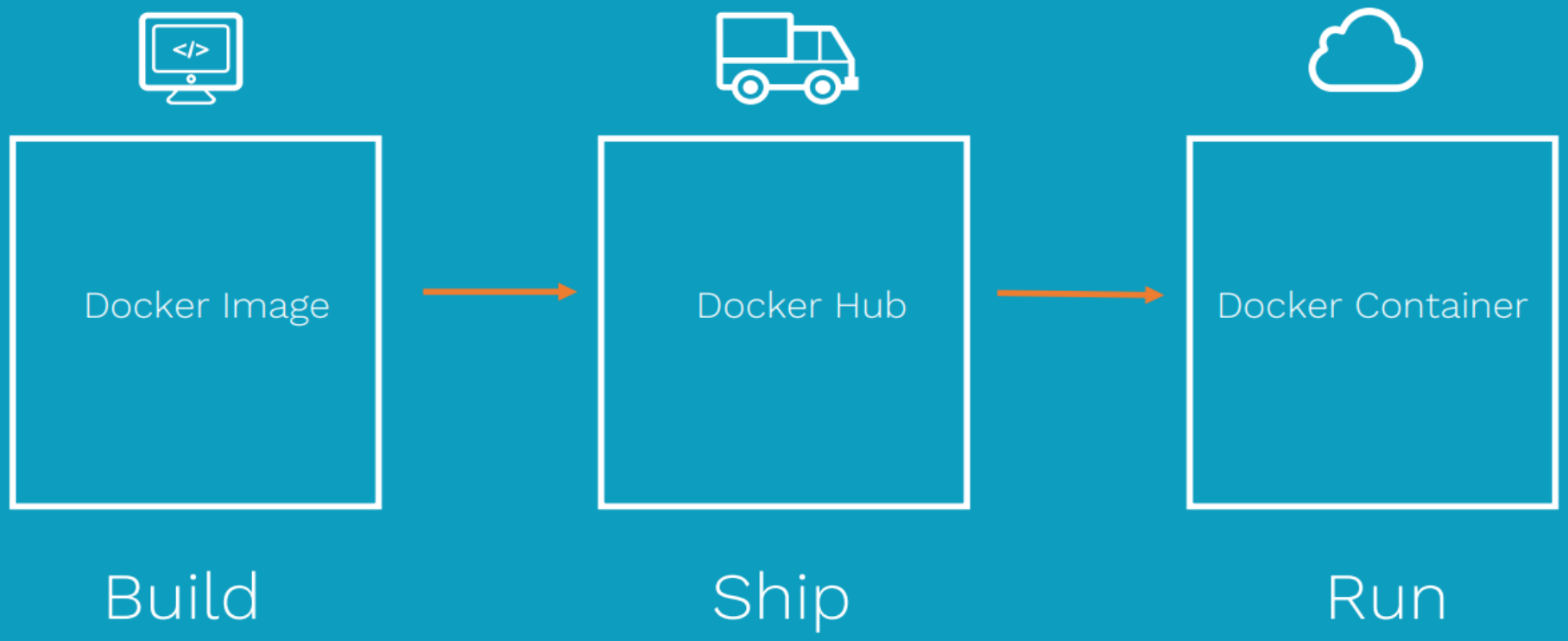
Package
Template
Plan

Docker Container #1

Docker Container #2

Docker Container #3

What it does?



Docker Hub

https://hub.docker.com/explore/

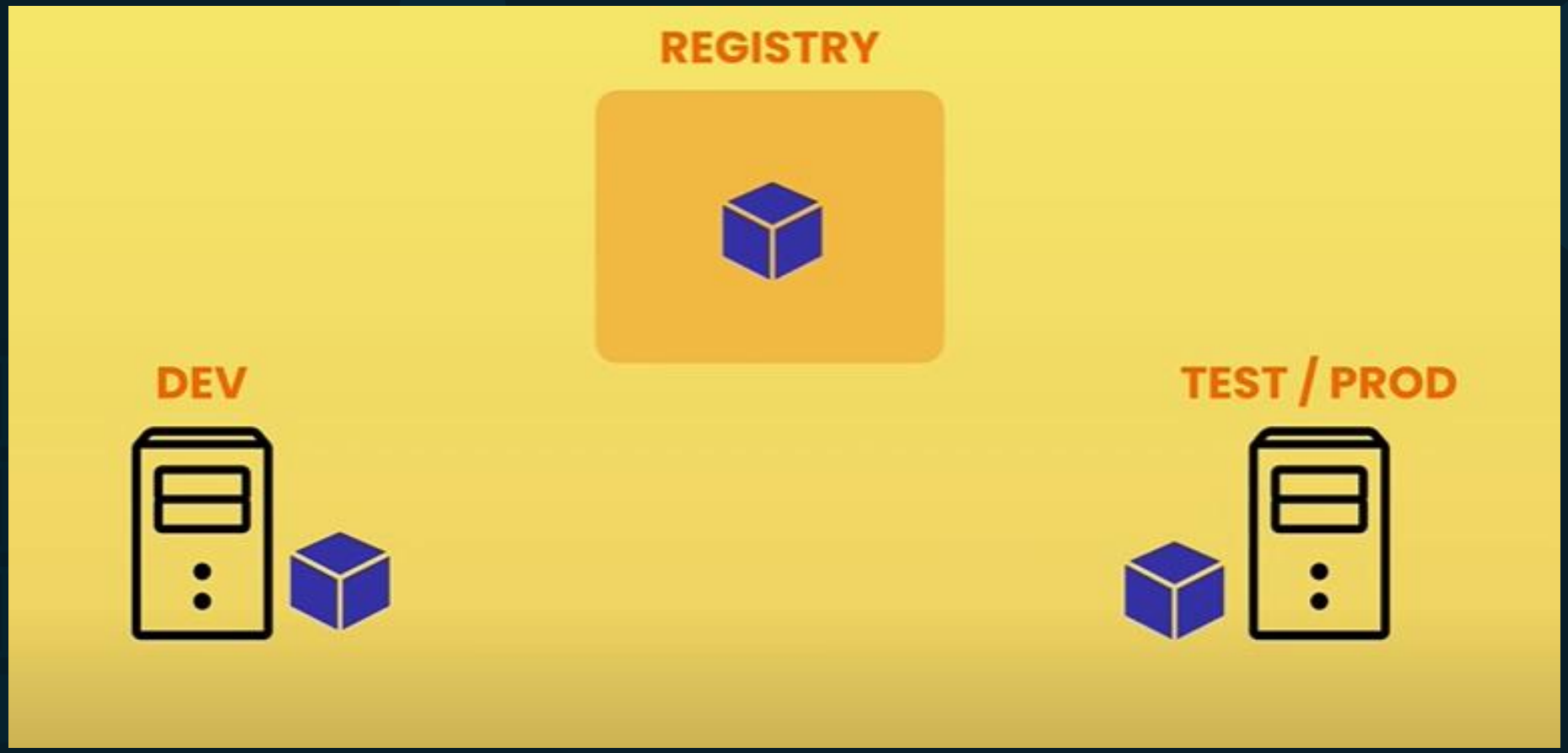
Search

Dashboard Explore Organizations Create srinathrchalla

Explore Official Repositories

nginx official	9.3K STARS	10M+ PULLS	DETAILS
alpine official	4.1K STARS	10M+ PULLS	DETAILS
busybox official	1.3K STARS	10M+ PULLS	DETAILS
httpd official	1.9K STARS	10M+ PULLS	DETAILS

Development Workflow

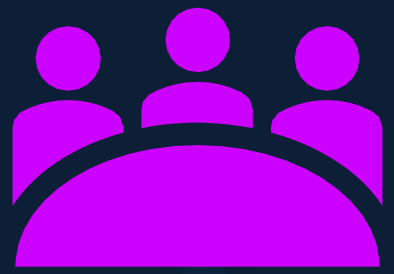


WISSEN

d o c k e r

Getting Started

Docker Editions



Community Edition

Enterprise Edition

Community Edition

Linux

MAC

Windows

Cloud

WISSEN

d o c k e r

On Windows

Docker on windows

1. Docker on Windows using Docker Toolbox
2. Docker Desktop for Windows

1. Docker toolbox

- 64-bit operating
- Windows 7 or higher.
- Virtualization is enabled

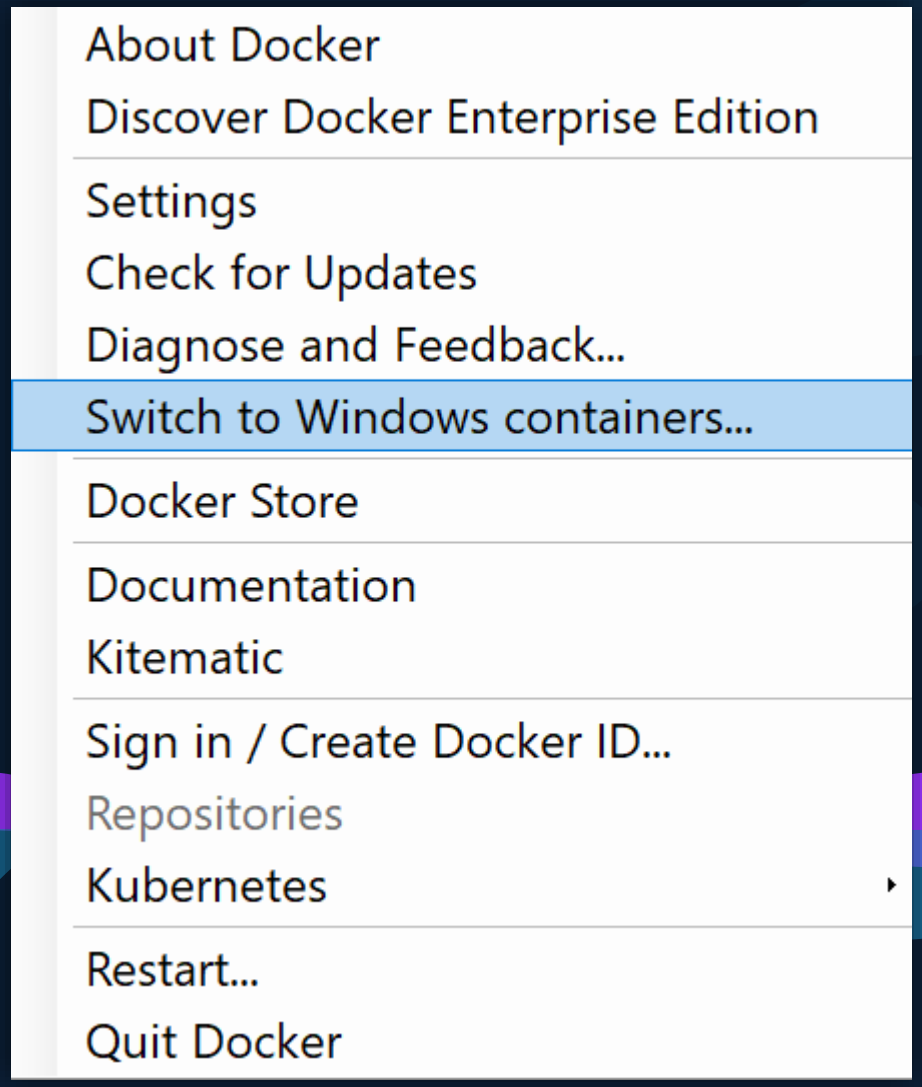
- Oracle Virtualbox
- Docker Engine
- Docker Machine
- Docker Compose
- Kitematic GUI

2. Docker Desktop for Windows

Microsoft
Hyper-V

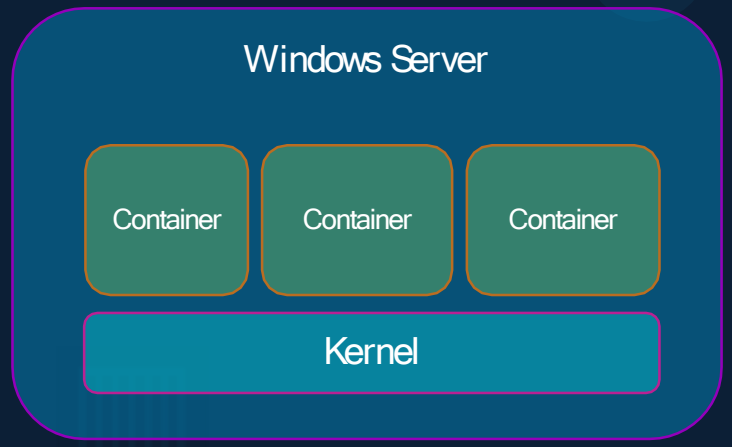
Support: Windows 10 Enterprise/Professional Edition
Windows Server 2016

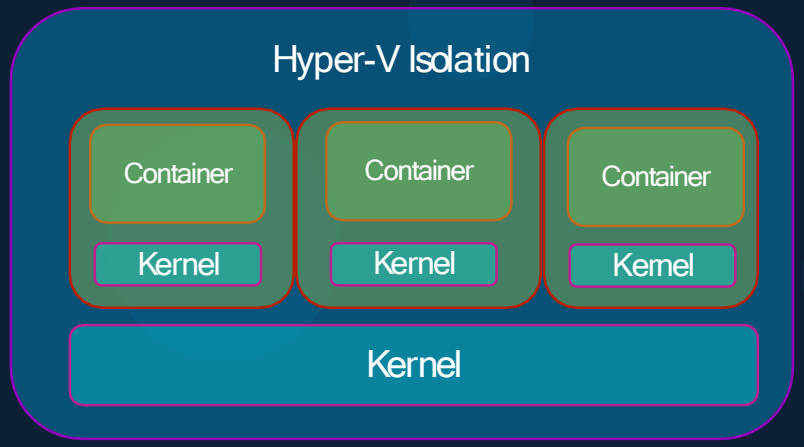
Linux Containers (Default)
Or
Windows Containers



Windows containers

Container Types:





Base Images:

- Windows Server Core
- Nano Server

Support

- Windows Server 2016
- Nano Server
- Windows 10 Professional and Enterprise (Hyper-V Isolated Containers)

VirtualBox Or Hyper-V

WISSEN

d o c k e r

On Mac

Docker on Mac

1. Docker on Mac using Docker Toolbox
2. Docker Desktop for Mac

1. Docker toolbox

- macOS 10.8 “Mountain Lion” or newer

- Oracle Virtualbox
- Docker Engine
- Docker Machine
- Docker Compose
- Kitematic GUI

2. Docker Desktop for Mac

HyperKit

Support: macOSSierra 10.12 or newer
Mac Hardware - 2010 model or newer

Linux Containers

WISSEN

docker

commands

Run – start a container

```
docker run nginx
```

```
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
fc7181108d40: Already exists
d2e987ca2267: Pull complete
0b760b431b11: Pull complete
Digest:
sha256:96fb261b66270b900ea5a2c17a26abbfabe95506e73c3a3c65869a6dbe83223a
Status: Downloaded newer image for nginx:latest
```

ps – list containers

```
docker ps
CONTAINER ID   IMAGE   COMMAND                  CREATED          STATUS          PORTS     NAMES
796856ac413d   nginx   "nginx -g 'daemon of…"   7 seconds ago    Up 6 seconds    80/tcp    silly_sammet
```

```
docker ps -a
CONTAINER ID   IMAGE   COMMAND                  CREATED          STATUS                     NAMES
796856ac413d   nginx   "nginx -g 'daemon of…"   7 seconds ago    Up 6 seconds               silly_sammet
cff8ac918a2f   redis   "docker-entrypoint.s…"   6 seconds ago    Exited (0) 3 seconds ago   relaxed_aryabhata
```

STOP – stop a container

```
docker ps
CONTAINER ID   IMAGE   COMMAND                  CREATED          STATUS          PORTS     NAMES
796856ac413d   nginx   "nginx -g 'daemon of…"   7 seconds ago    Up 6 seconds    80/tcp    silly_sammet
```

```
docker stop silly_sammet
silly_sammet
```

```
CONTAINER ID   IMAGE   COMMAND                  CREATED          STATUS                      NAMES
cff8ac918a2f   redis   "docker-entrypoint.s…"   6 seconds ago    Exited (0) 3 seconds ago    relaxed_aryabhata
```

Rm – Remove a container

```
docker rm silly_sammet
```

```
silly_sammet
```

```
docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	NAMES
cff8ac918a2f	redis	"docker-entrypoint.s…"	6 seconds ago	Exited (0) 3 seconds ago	relaxed_aryabhata

images – List images

```
docker images
```

```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nginx               latest              f68d6e55e065        4 days ago          109MB
redis               latest              4760dc956b2d        15 months ago       107MB
ubuntu              latest              f975c5035748        16 months ago       112MB
alpine              latest              3fd9065eaf02        18 months ago       4.14MB
```

rmi – Remove images

```
docker rmi nginx
```

```
Untagged: nginx:latest
Untagged: nginx@sha256:96fb261b66270b900ea5a2c17a26abbfabe95506e73c3a3c65869a6dbe83223a
Deleted: sha256:f68d6e55e06520f152403e6d96d0de5c9790a89b4cfc99f4626f68146fa1dbdc
Deleted: sha256:1b0c768769e2bb66e74a205317ba531473781a78b77feef8ea6fd7be7f4044e1
Deleted: sha256:34138fb60020a180e512485fb96fd42e286fb0d86cf1fa2506b11ff6b945b03f
Deleted: sha256:cf5b3c6798f77b1f78bf4e297b27cfa5b6caa982f04caeb5de7d13c255fd7a1e
```

! Delete all dependent containers to remove image

Pull – download an image

```
docker run nginx
```

```
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
fc7181108d40: Already exists
d2e987ca2267: Pull complete
0b760b431b11: Pull complete
Digest:
sha256:96fb261b66270b900ea5a2c17a26abbfabe95506e73c3a3c65869a6dbe83223a
Status: Downloaded newer image for nginx:latest
```

```
docker pull nginx
```

```
Using default tag: latest
latest: Pulling from library/nginx
fc7181108d40: Pull complete
d2e987ca2267: Pull complete
0b760b431b11: Pull complete
Digest:
sha256:96fb261b66270b900ea5a2c17a26abbfabe95506e73c3a3c65869a6dbe83223a
Status: Downloaded newer image for nginx:latest
```

Status of Container

```
docker run ubuntu
```

```
docker ps
```

```
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
```

```
docker ps -a
```

```
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                      PORTS
45aacca36850        ubuntu              "/bin/bash"         43 seconds ago      Exited (0) 41 seconds ago
```

Run command

```
docker run ubuntu
```

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS

```
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
45aacca36850	ubuntu	"/bin/bash"	43 seconds ago	Exited (0) 41 seconds ago	

Append a command

```
docker run ubuntu
```

```
docker run ubuntu sleep 5
```

Exec – execute a command

```
docker ps -a
```

```
CONTAINER ID        IMAGE     COMMAND       CREATED             STATUS              NAMES
538d037f94a7        ubuntu    "sleep 100“   6 seconds ago       Up 4 seconds        distracted_mcclintock
```

```
docker exec distracted_mcclintock 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
172.18.0.2      538d037f94a7
```

Run – attach and detach

```
docker run knagendra0521/simple-webapp
```

```
This is a sample web application that displays a colored background.
* Serving Flask app "app" (lazy loading)
* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
```

```
docker run -i knagendra0521/simple-webapp
```

```
a043d40f85fefa414254e4775f9336ea59e19e5cf597af5c554e0a35a1631118
```

```
docker attach a043d
```

WISSEN

docker run

Run – tag

```
docker run redis
```

```
Using default tag: latest
latest: Pulling from library/redis
f5d23c7fed46: Pull complete
Status: Downloaded newer image for redis:latest

1:C 31 Jul 2019 09:02:32.624 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
1:C 31 Jul 2019 09:02:32.624 # Redis version=5.0.5, bits=64, commit=00000000, modified=0, pid=1, just started
1:M 31 Jul 2019 09:02:32.626 # Server initialized
```

```
docker run redis:4.0
```

TAG

```
Unable to find image 'redis:4.0' locally
4.0: Pulling from library/redis
e44f086c03a2: Pull complete
Status: Downloaded newer image for redis:4.0

1:C 31 Jul 09:02:56.527 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
1:C 31 Jul 09:02:56.527 # Redis version=4.0.14, bits=64, commit=00000000, modified=0, pid=1, just started
1:M 31 Jul 09:02:56.530 # Server initialized
```

RUN - STDIN

```
~/prompt-application$ ./app.sh
Welcome! Please enter your name: Nagendra

Hello and Welcome Nagendra!
```

```
docker run knagendra0521/simple-prompt-docker
```

```
Hello and Welcome !
```

```
docker run -i knagendra-0521/simple-prompt-docker
```

```
Nagendra

Hello and Welcome Nagendra!
```

```
docker run -it knagendra0521/simple-prompt-docker
```

```
Welcome! Please enter your name: Nagendra

Hello and Welcome Nagendra!
```

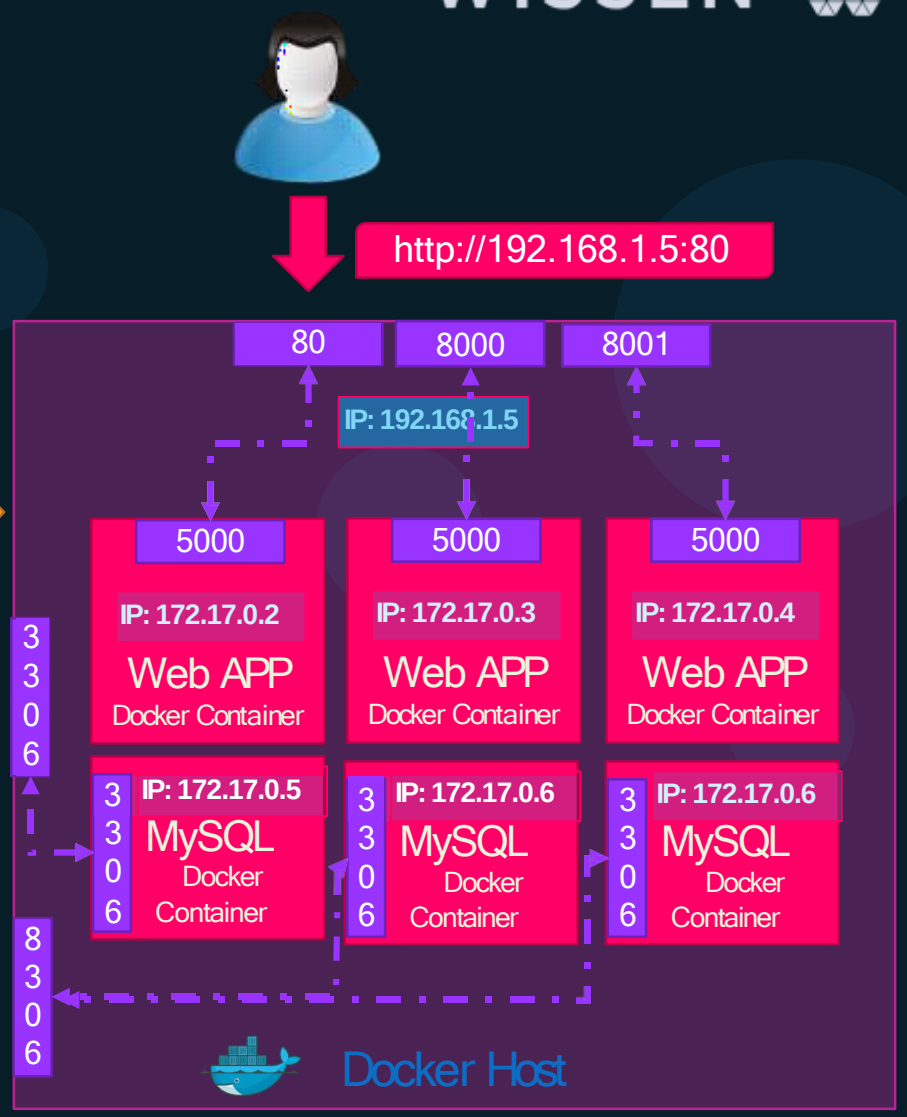
Run – PORT mapping

```
docker run knagendra0521/webapp
```

```
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

http://172.17.0.2:5000

Internal IP




```
docker run -p 80:5000 knagendra0521/simple-webapp
```

```
docker run -p 8000:5000 knagendra0521/simple-webapp
```

```
docker run -p 8001:5000 knagendra0521/simple-webapp
```

```
docker run -p 3306:3306 mysql
```

```
docker run -p 8306:3306 mysql
```

```
docker run -p 8306:3306 mysql
```

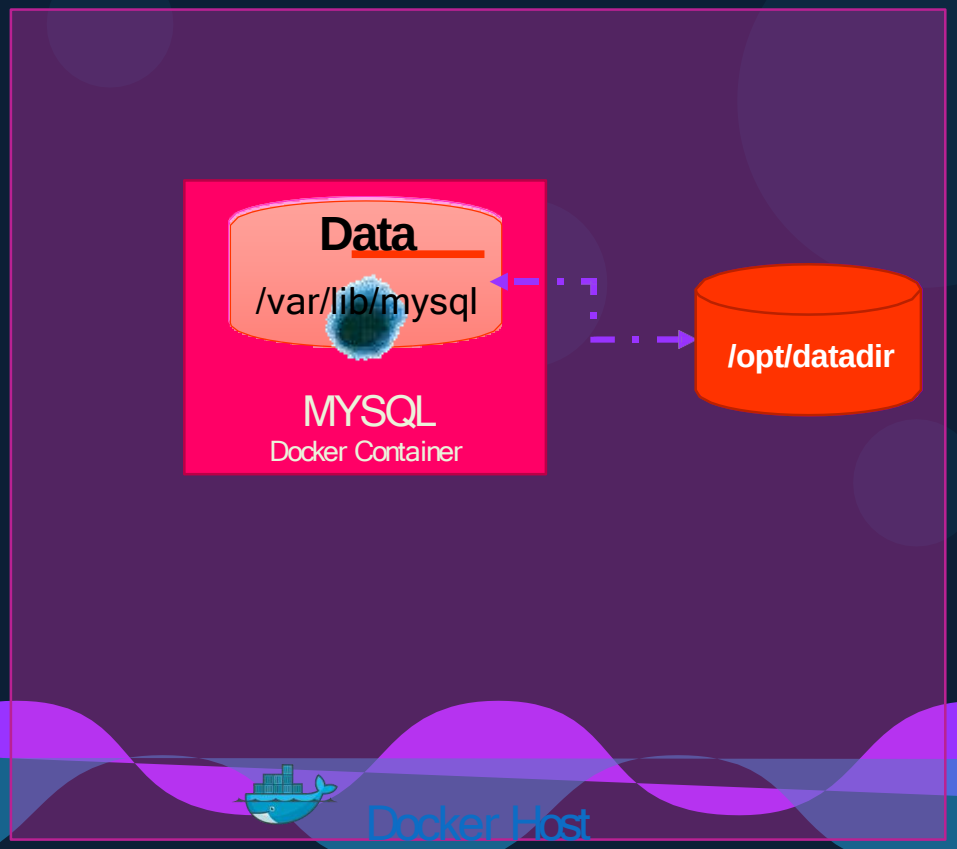
```
root@osboxes:/root # docker run -p 8306:3306 -e MYSQL_ROOT_PASSWORD=pass mysql
docker: Error response from daemon: driver failed programming external connectivity on endpoint boring_bhabha (
5079d342b7e8ee11c71d46): Bind for 0.0.0.0:8306 failed: port is already allocated.
```

RUN – Volume mapping

```
docker run mysql
```

```
docker stop mysql
docker rm mysql
```

```
docker run -v /opt/datadir:/var/lib/mysql mysql
```



Inspect Container

```
docker inspect blissful_hopper
```


```
[
    {
        "Id": "35505f7810d17291261a43391d4b6c0846594d415ce4f4d0a6ffbf9cc5109048",
        "Name": "/blissful_hopper",
        "Path": "python",
        "Args": [
            "app.py"
        ],
        "State": {
            "Status": "running",
            "Running": true,
        },

        "Mounts": [],
        "Config": {
           "Entrypoint": [
                "python",
                "app.py"
            ],
        },
        "NetworkSettings": {..}
      }
]
```

Container Logs

```
docker logs blissful_hopper
```

```
 This is a sample web application that displays a colored background.
 A color can be specified in two ways.

 1. As a command line argument with --color as the argument. Accepts one of
red,green,blue,blue2,pink,darkblue
2.As an Environment variable APP_COLOR. Accepts one of
red,green,blue,blue2,pink,darkblue
 3. If none of the above then a random color is picked from the above list.
 Note: Command line argument precedes over environment variable.


No command line argument or environment variable. Picking a Random Color =blue
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
```

What can you Containerize?



Containerize Everything!!!

WISSEN

docker

CMD vs ENTRYPOINT

```
docker run ubuntu [COMMAND]
```

```
docker run ubuntu sleep 5
```

```
FROM Ubuntu

CMD sleep 5
```

CMD command param1

CMD [“command”, “param1”]

CMD sleep 5

CMD [“sleep”, “5”]

CMD [“sleep 5”]

```
docker build -t ubuntu-sleeper .
```

```
docker run ubuntu-sleeper
```

```
FROM Ubuntu

CMD sleep 5
```

```
docker run ubuntu-sleeper sleep 10
```

Command at Startup: sleep 10

```
FROM Ubuntu

ENTRYPOINT [“sleep”]
```

```
docker run ubuntu-sleeper 10
```

Command at Startup:

```
docker run ubuntu-sleeper
```

```
sleep: missing operand
Try 'sleep --help' for more information.
```

Command at Startup:

FROM Ubuntu

ENTRYPOINT ["sleep"]

CMD ["5"]

```
docker run ubuntu-sleeper
```

```
sleep: missing operand
Try 'sleep --help' for more information.
```

Command at Startup:

```
docker run ubuntu-sleeper 10
```

Command at Startup:

```
docker run --entrypoint sleep2.0 ubuntu-sleeper 10
```

Command at Startup:

WISSEN

WISSEN

d o c k e r
networking

Default networks

WISSEN

Bridge

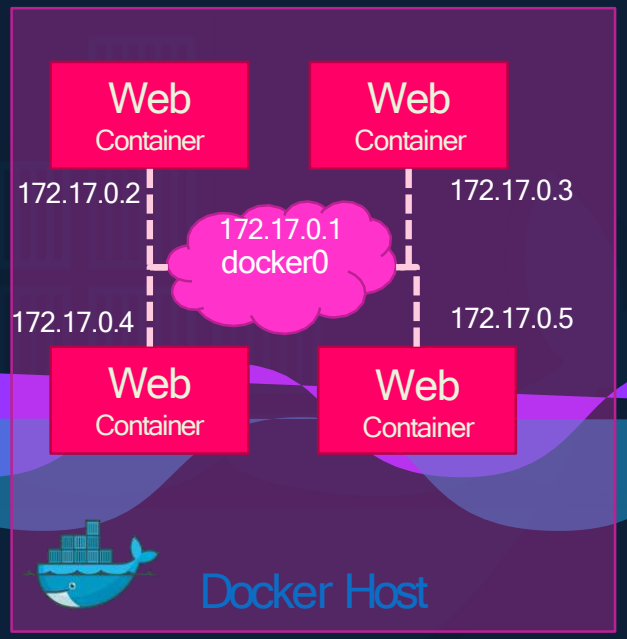
```
docker run ubuntu
```

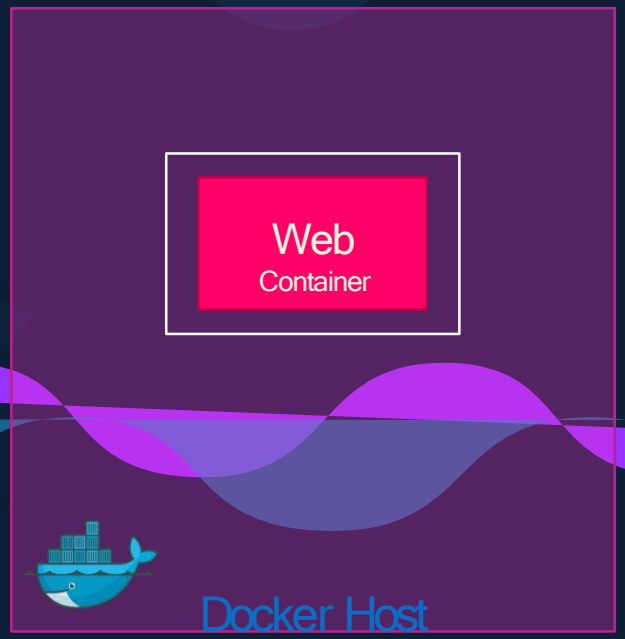
none

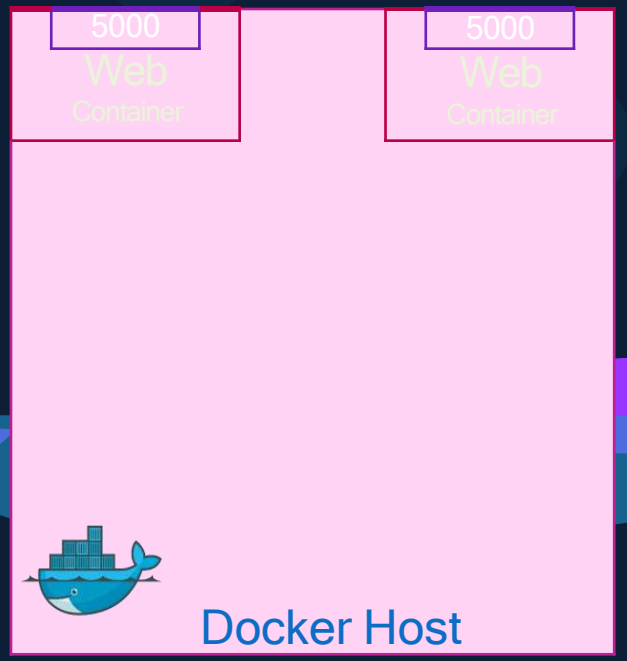
```
docker run Ubuntu --network=none
```

host

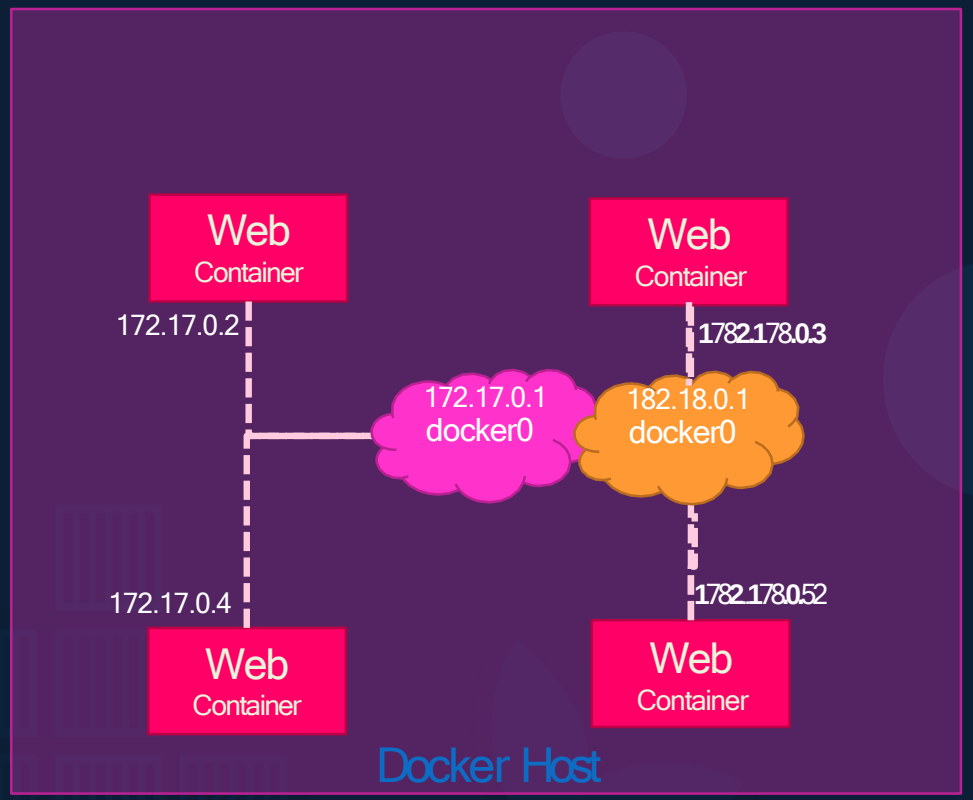
```
docker run Ubuntu --network=host
```







User-defined networks




```
docker network create \
    --driver bridge \
    --subnet 182.18.0.0/16
    custom-isolated-network
```

```
docker network ls
```

```
root@osboxes:/root # docker network ls
NETWORK ID          NAME                        DRIVER              SCOPE
dba0fb9370fe        bridge                      bridge              local
46d476b87cd9        customer-isolated-network   bridge              local
6de685cec1ce        docker_gwbridge             bridge              local
e29d188b4e47        host                        host                local
mmrho7vsb9rm        ingress                     overlay             swarm
d9f11695f0d6        none                        null                local
d371b4009142        simplewebappdocker_default  bridge              local
```

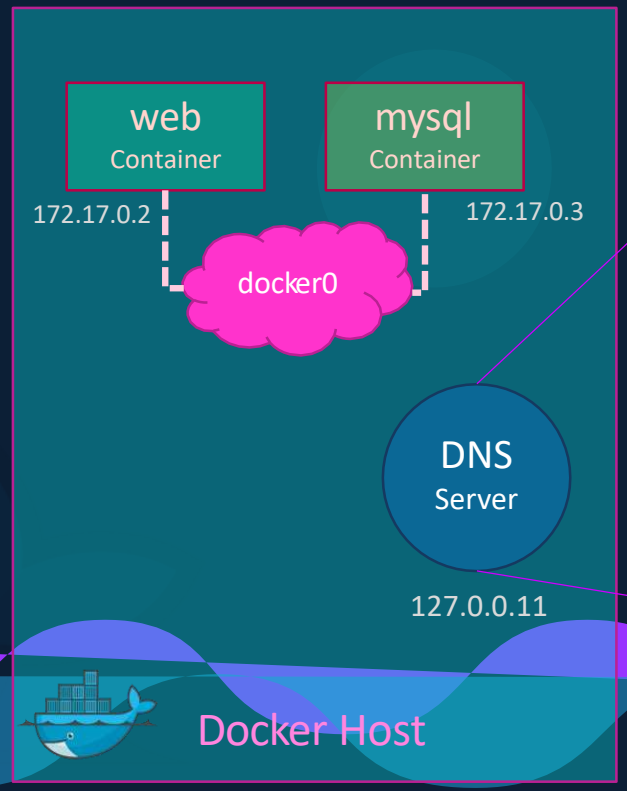
Inspect Network

```
docker inspect blissful_hopper
```

```
[
    {
        "Id": "35505f7810d17291261a43391d4b6c0846594d415ce4f4d0a6ffbf9cc5109048",
        "Name": "/blissful_hopper",
        "NetworkSettings": {
            "Bridge": "",
            "Gateway": "172.17.0.1",
            "IPAddress": "172.17.0.6",
            "MacAddress": "02:42:ac:11:00:06",
            "Networks": {
                "bridge": {
                    "Gateway": "172.17.0.1",
                    "IPAddress": "172.17.0.6",
                    "MacAddress": "02:42:ac:11:00:06",
                }
            }
        }
]
```

Embedded DNS

```
mysql.connect(   mysql      )
```



Host	IP
web	172.17.0.2
mysql	172.17.0.3

WISSEN

docker storage

File system

- /var/lib/docker
  - aufs
  - containers
  - image
  - volumes

Layered architecture

Container Layer

Read Write

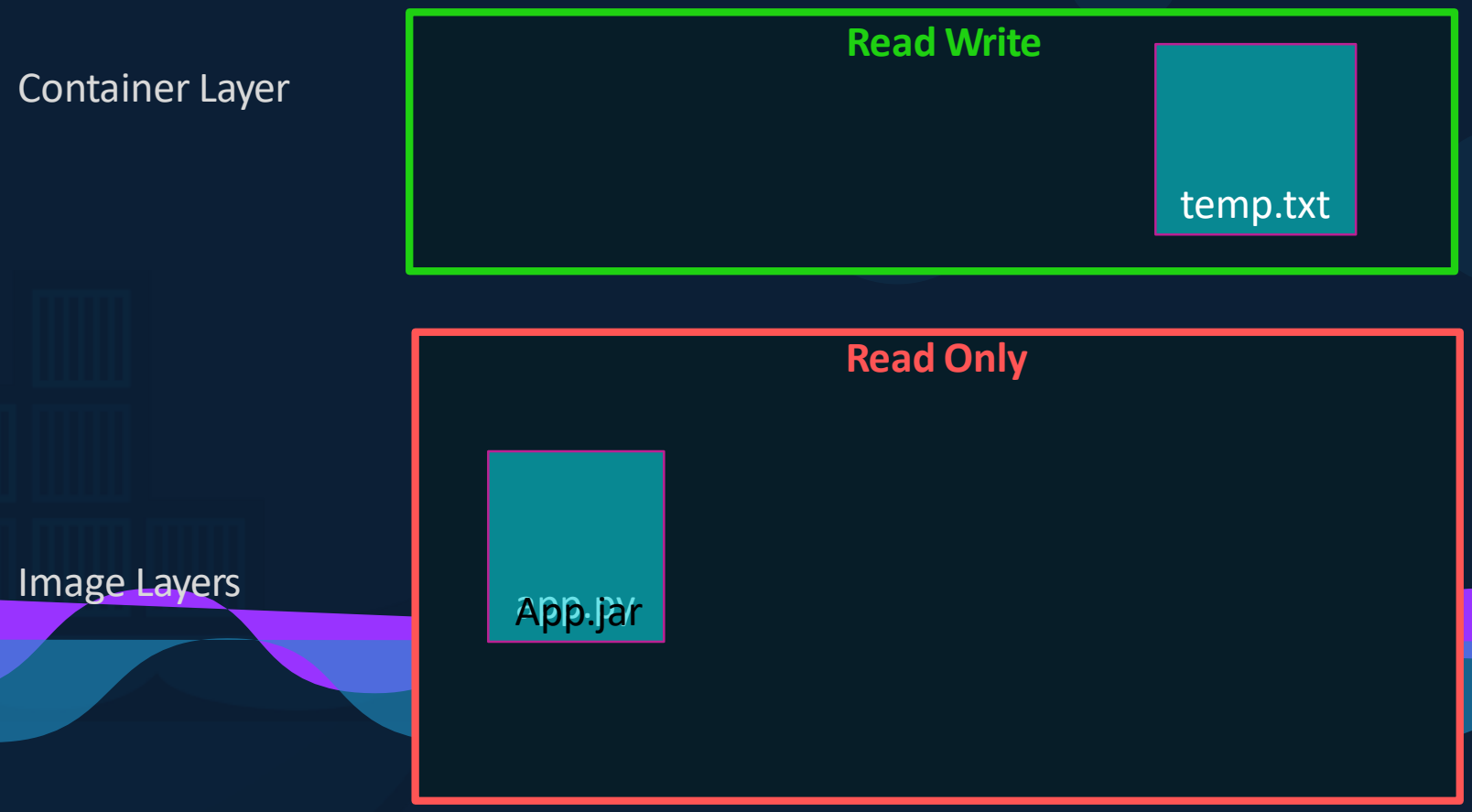
```
docker run knagendra0521/my-custom-app
```

Image Layers

Read Only

```
docker build Dockerfile -t knagendra0521/my-custom-app
```

COPY-ON-WRITE



volumes

```
docker volume create data_volume
```

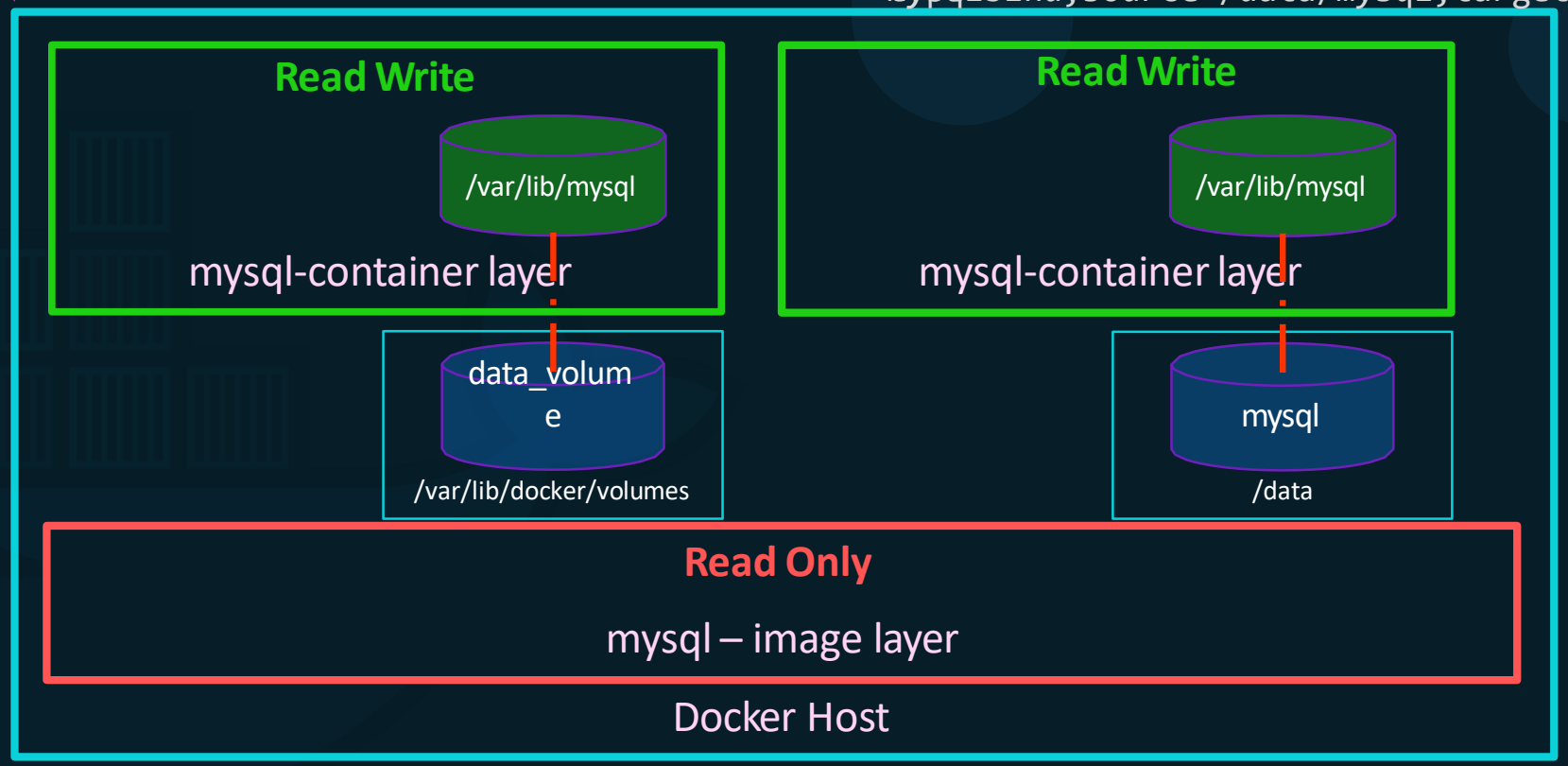
- /var/lib/docker
  - volumes
    - data_volume

```
docker run -v data_volume:/var/lib/mysql mysql
```

```
docker run -v data_volume2:/var/lib/mysql mysql
```

```
docker run -v /data/mysql:/var/lib/mysql mysql
```

```
docker run \
--mount
type=bind,source=/data/mysql,target=/var/lib/mysql
mysql
```



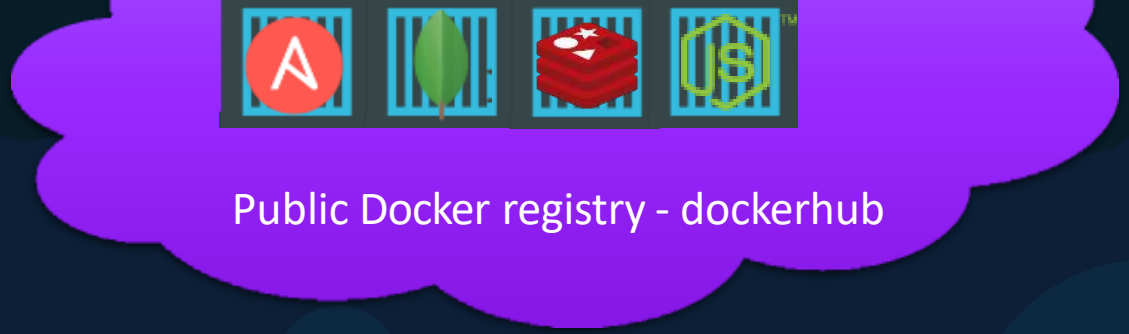
Storage drivers

- AUFS
- ZFS
- BTRFS
- Device Mapper
- Overlay
- Overlay2

WISSEN

docker compose

Docker compose




```
docker run knagendra0521/simple-webapp
```

```
docker run mongodb
```

```
docker run redis:alpine
```

```
docker run ansible
```

docker-compose.yml

```
services:
    web:
    image: “knagendra0521/simple-webapp"
  database:
    image: “mongodb“
  messaging:
    image: "redis:alpine“
  orchestration:
    image:  “ansible“
```

```
docker-compose up
```

WISSEN

Docker compose

docker-compose.yml

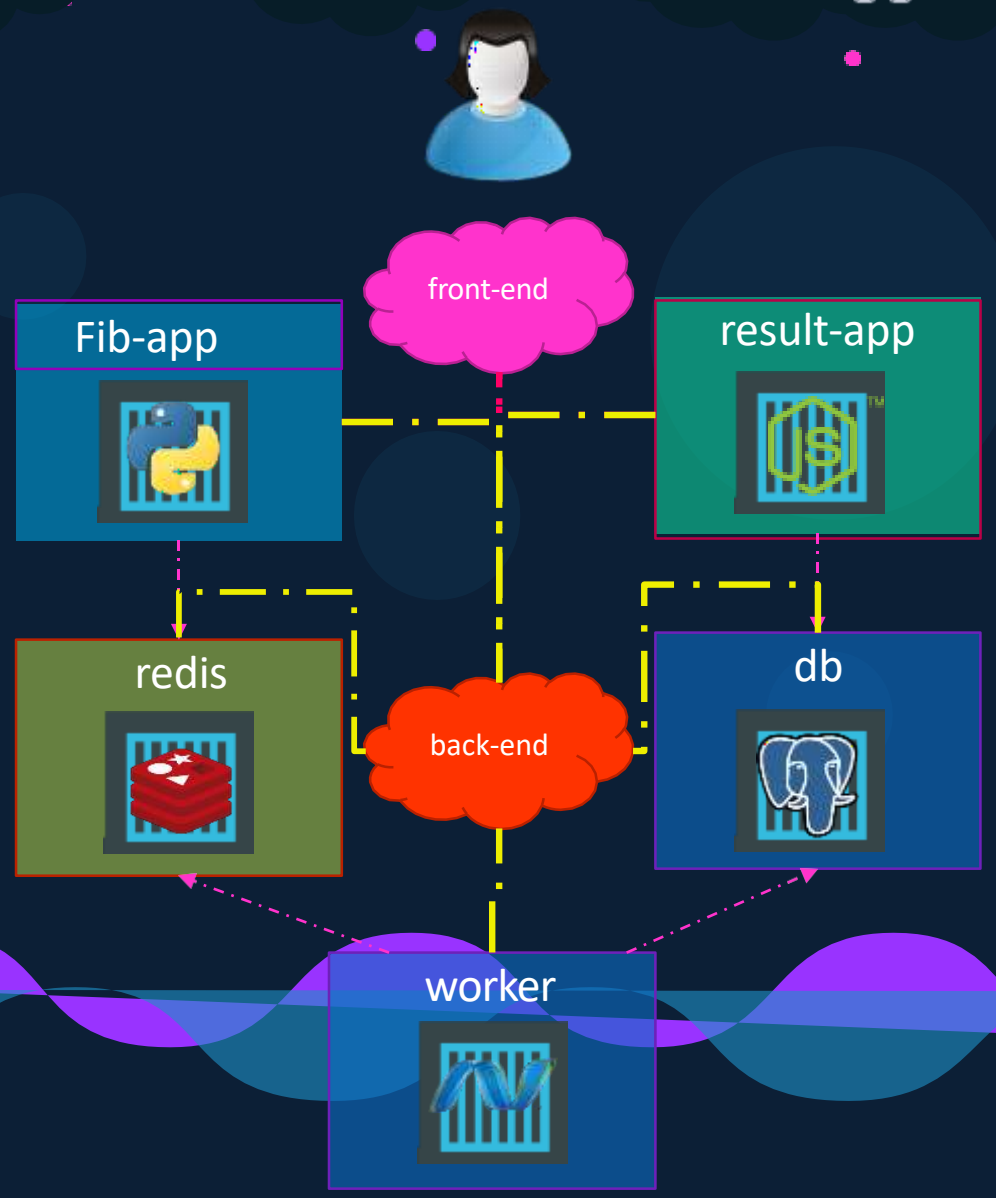
```yaml
version: 2
services:
    redis:
        image: redis

        networks:
            - back-end
    db:
        image: postgres:9.4
        networks:
            - back-end

    vote:
        image: voting-app
        networks:
            - front-end
            - back-end

    result:
        image: result
        networks:
            - front-end
            - back-end

networks:
    front-end:
    back-end:
```



WISSEN

docker registry

Image

```
docker run nginx
```

Image

docker.io
Docker Hub

WISSEN

image: `docker.io/nginx/nginx`

Registry | User/ Account | Image/ Repository

`gcr.io/ kubernetes-e2e-test-images/dnsutils`

Private Registry

```
docker login private-registry.io
```

```
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to
https://hub.docker.com to create one.
Username: registry-user
Password:
WARNING! Your password will be stored unencrypted in /home/vagrant/.docker/config.json.

Login Succeeded
```

```
docker run private-registry.io/apps/internal-app
```

Deploy Private Registry

```
docker run -d -p 5000:5000 --name registry registry:2
```

```
docker image tag my-image localhost:5000/my-image
```

```
docker push localhost:5000/my-image
```

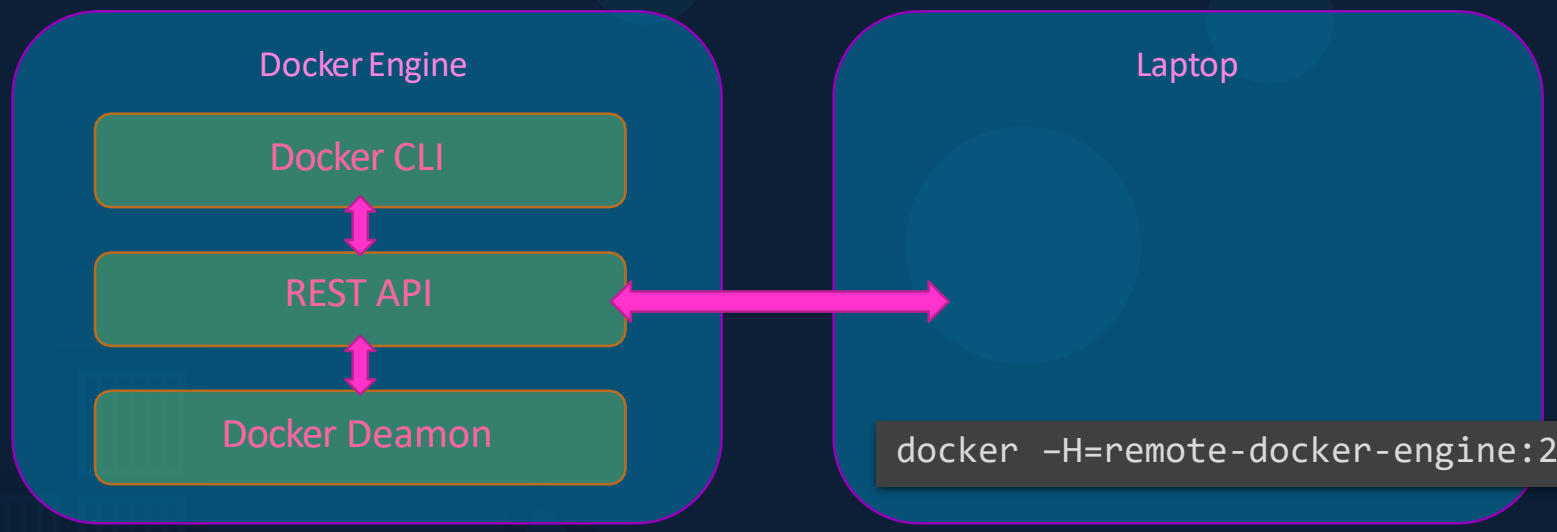
```
docker pull localhost:5000/my-image
```

```
docker pull 192.168.56.100:5000/my-image
```

WISSEN

docker engine

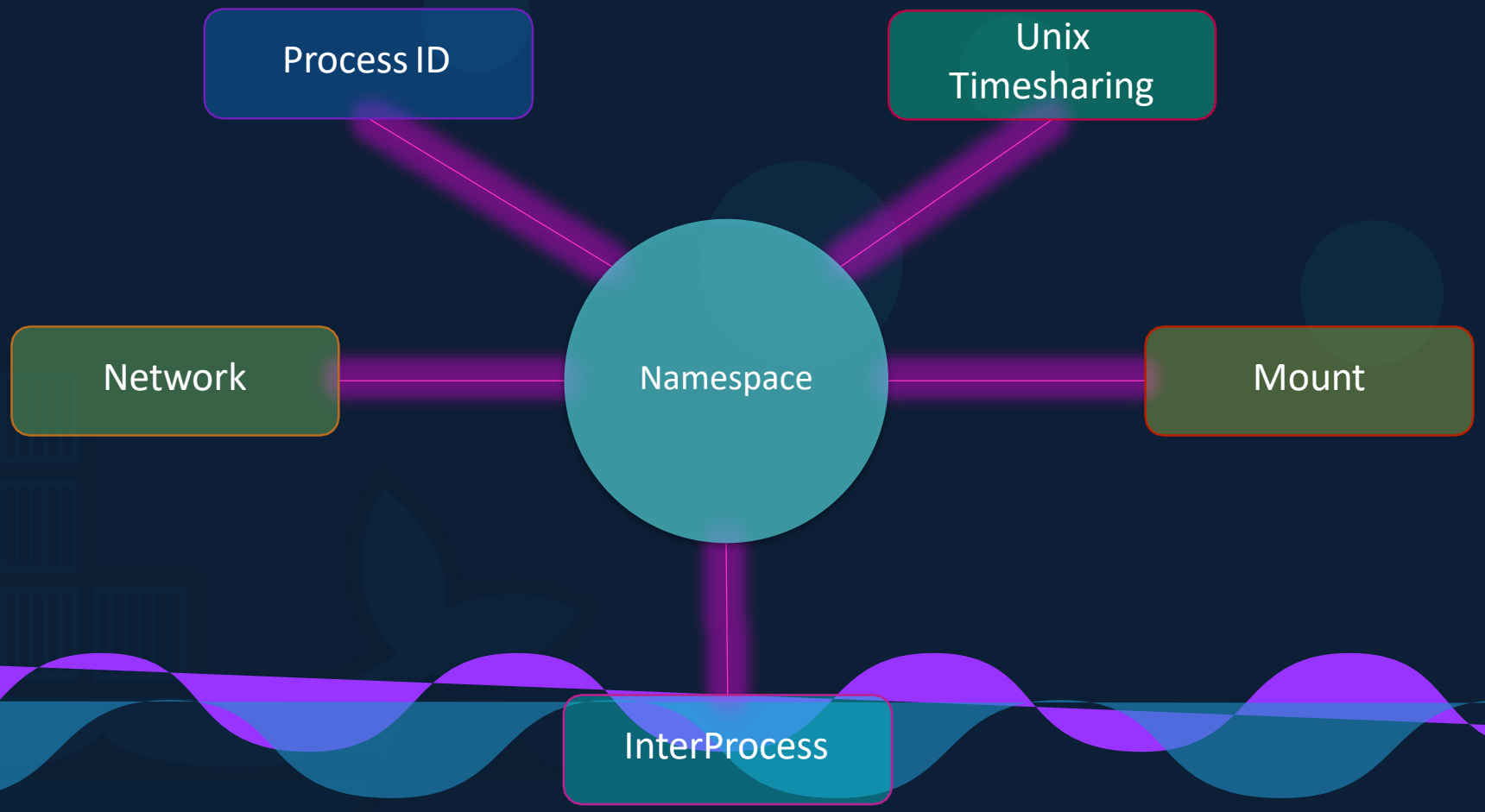
Docker Engine



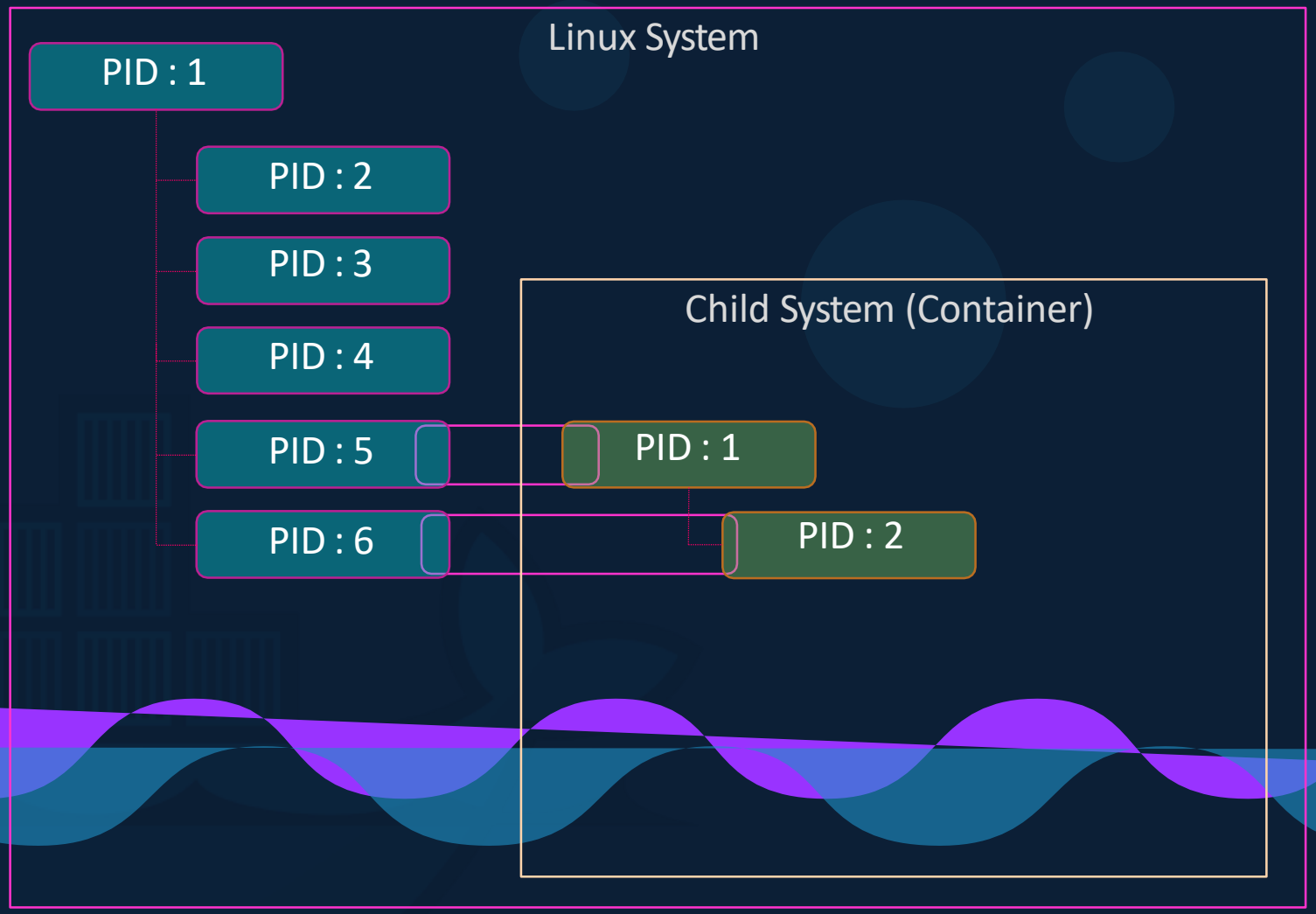

```
docker -H=remote-docker-engine:2375
```

```
docker -H=10.123.2.1:2375 run nginx
```

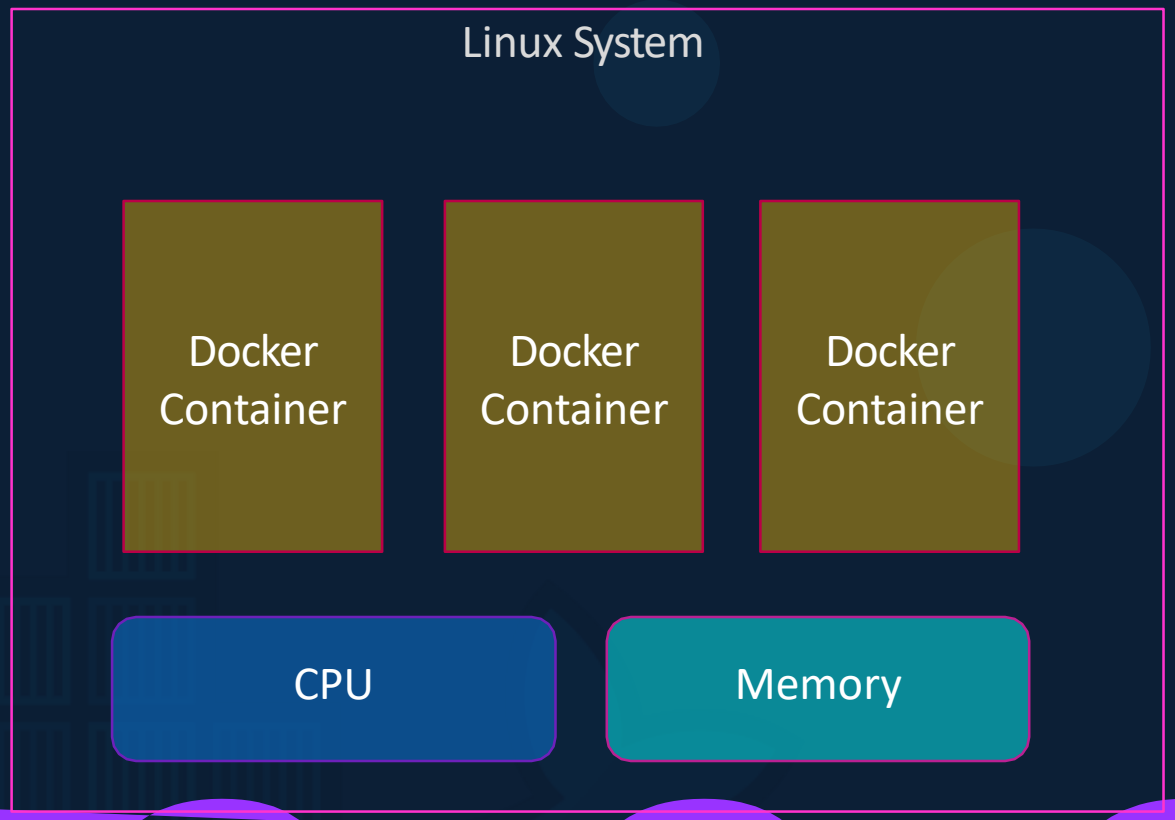
containerization



Namespace - PID



cgroups




```
docker run --cpus=.5 ubuntu
```

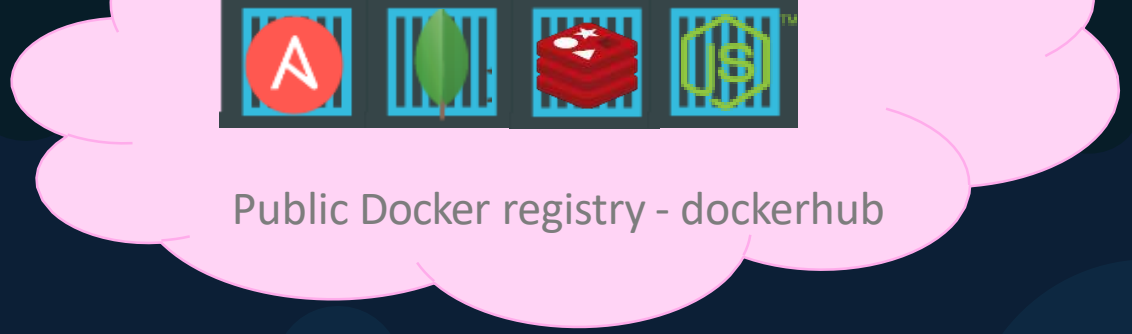
```
docker run --memory=100m ubuntu
```

WISSEN

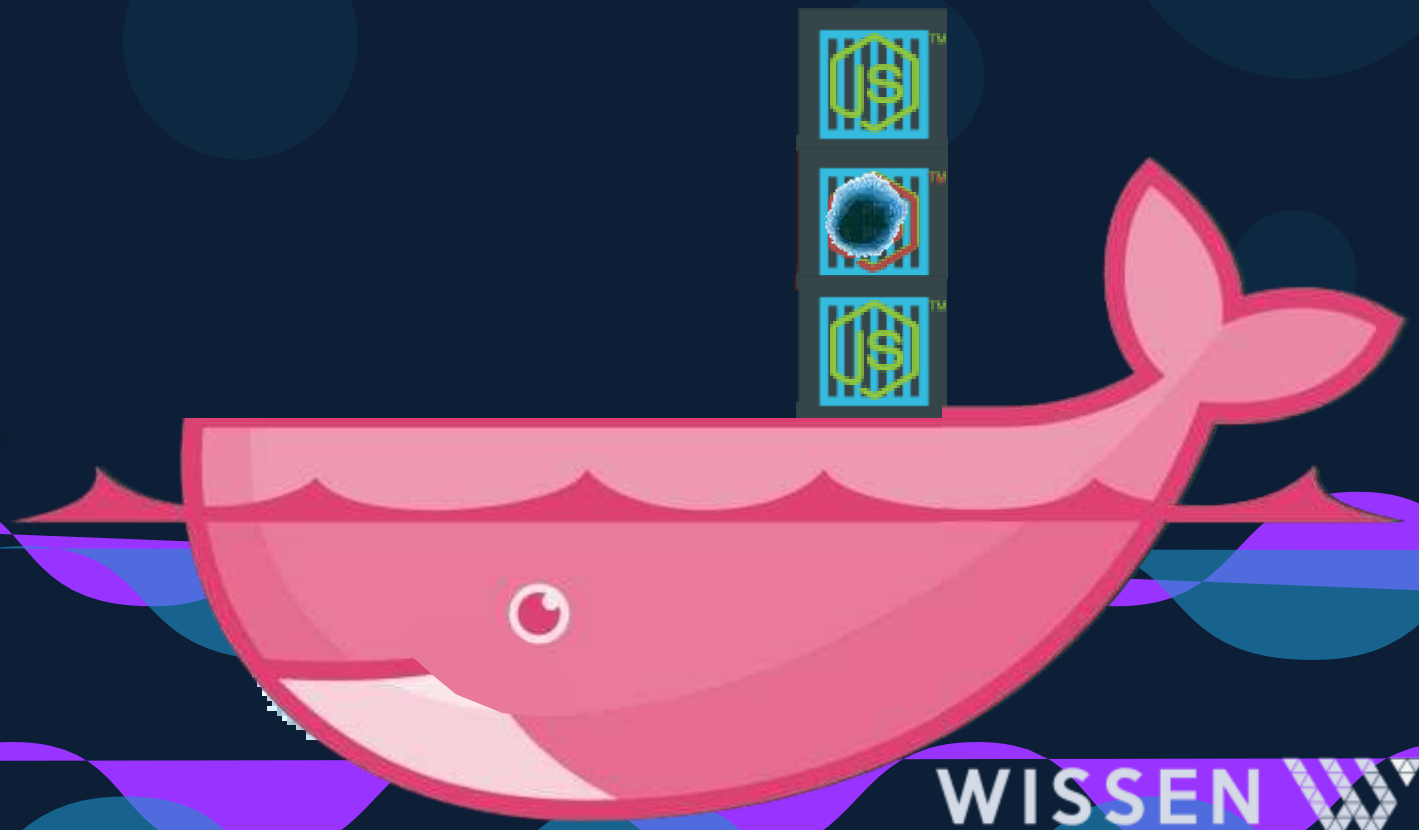
container

orchestration

Why Orchestrate?




```
docker run nodejs
docker run nodejs
docker run nodejs
docker run nodejs
```

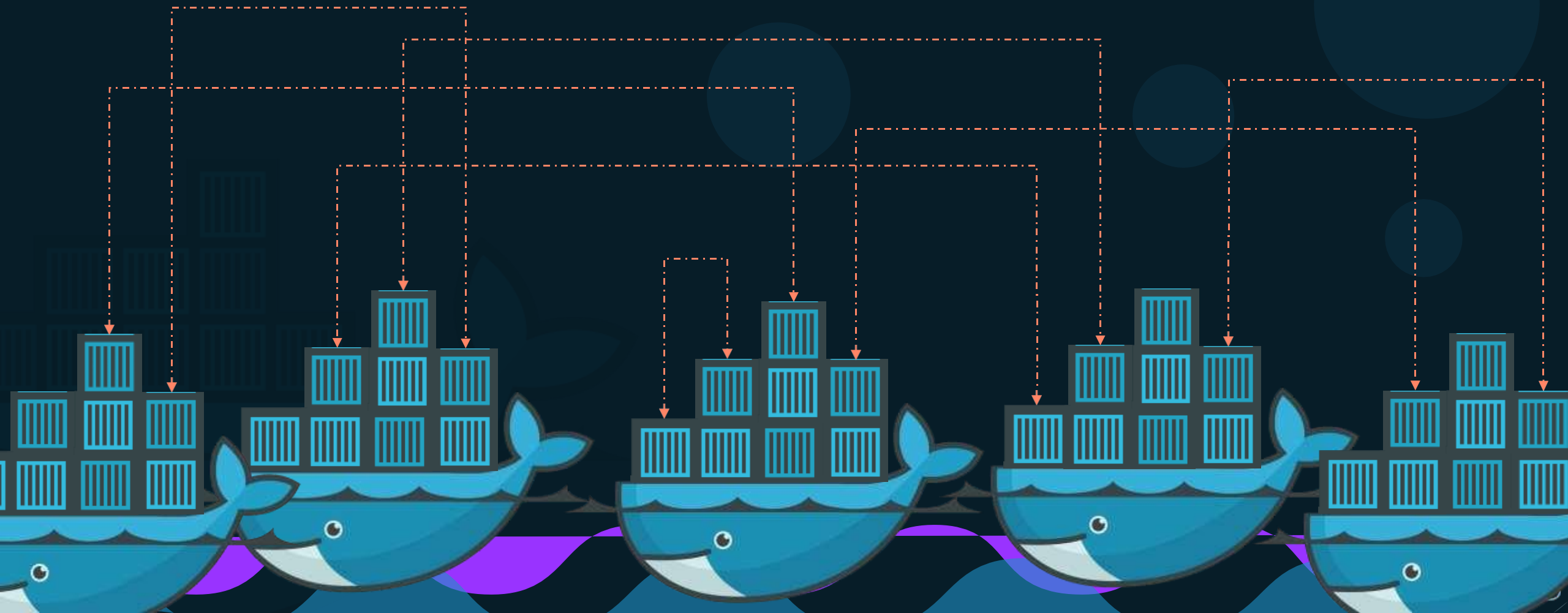


Container Orchestration

```
docker service create --replicas=100 nodejs
```

Container Orchestration

```
docker service create --replicas=100 nodejs
```



WISSEN

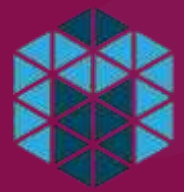
d o c k e r
swarm

Solutions

WISSEN

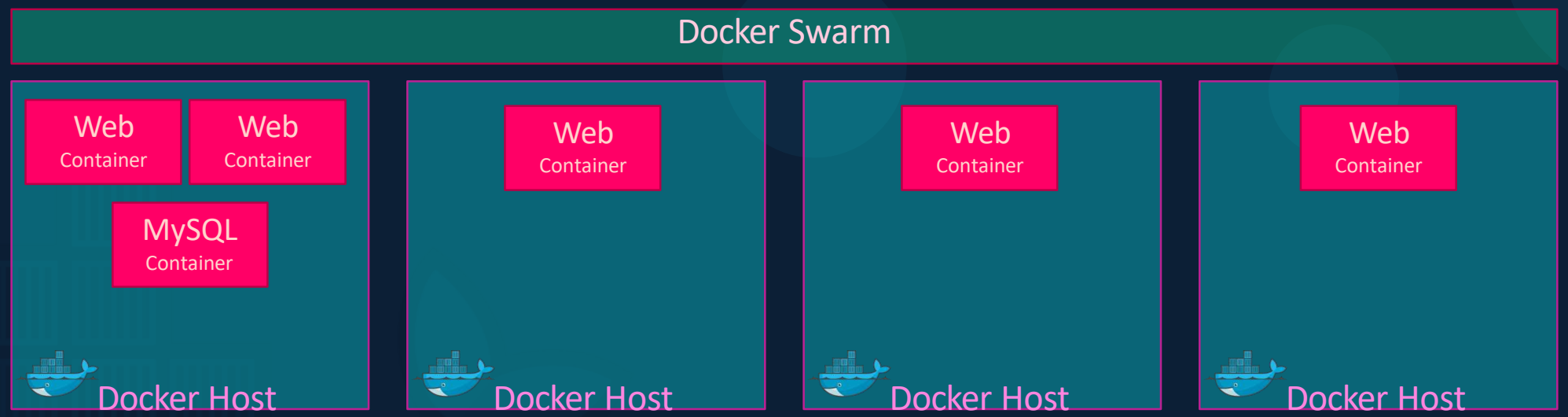
Docker Swarm

kubernetes



MESOS

Docker swarm



Setup swarm

WISSEN

Swarm Manager	Node Worker	Node Worker	Node Worker
`docker swarm init`	`docker swarm join --token <token>`	`docker swarm join --token <token>`	`docker swarm join --token <token>`
Docker Host	Docker Host	Docker Host	Docker Host

```
root@osboxes:/root/simple-webapp-docker # docker swarm init --advertise-addr 192.168.1.12
Swarm initialized: current node (0j76dum2r56p1xfne4ub1ps2c) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-35va8b3fi5krpdskefqqxgttmulw3z828daucri7y526ne0sgu-2eek9qm33d4lxzoq6we9i8izp 192.168.1.12:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

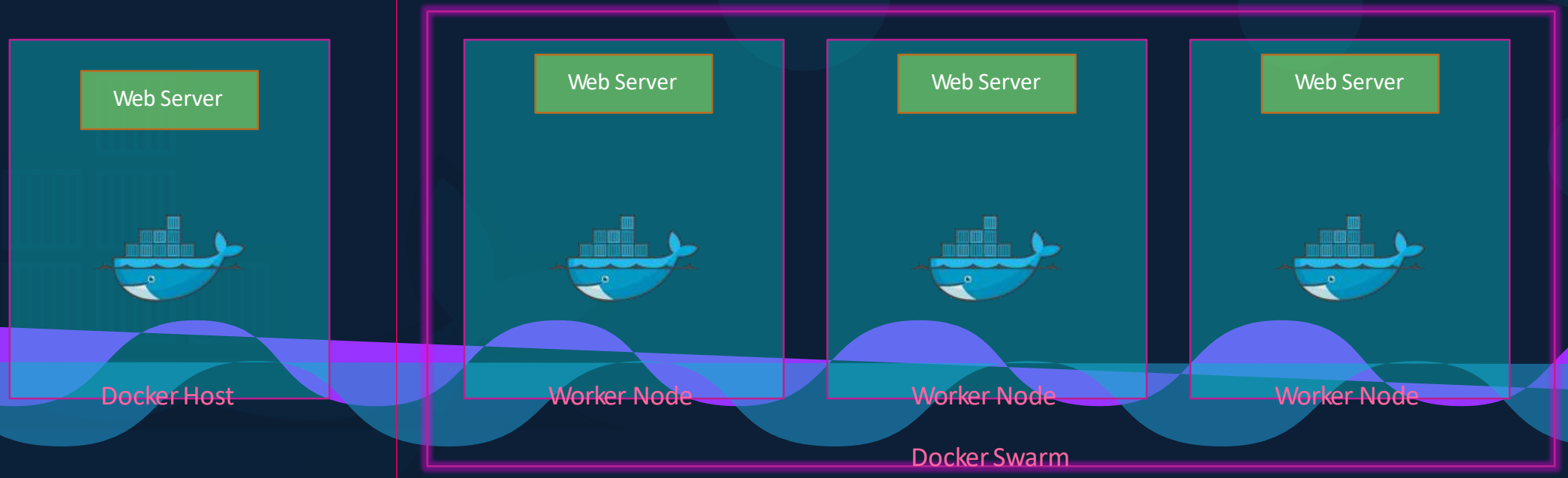
Docker service

```
docker run my-web-server
```

```
docker service create --replicas=3 --network frontend my-web-server
```

```
docker service create --replicas=3 -p 8080:80 my-web-server
```

```
docker service create --replicas=3 my-web-server
```



WISSEN

kubernetes

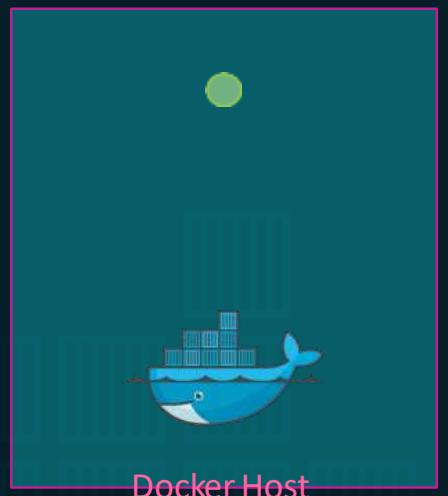
```
kubectl rolling-update my-web-server --rollback
```

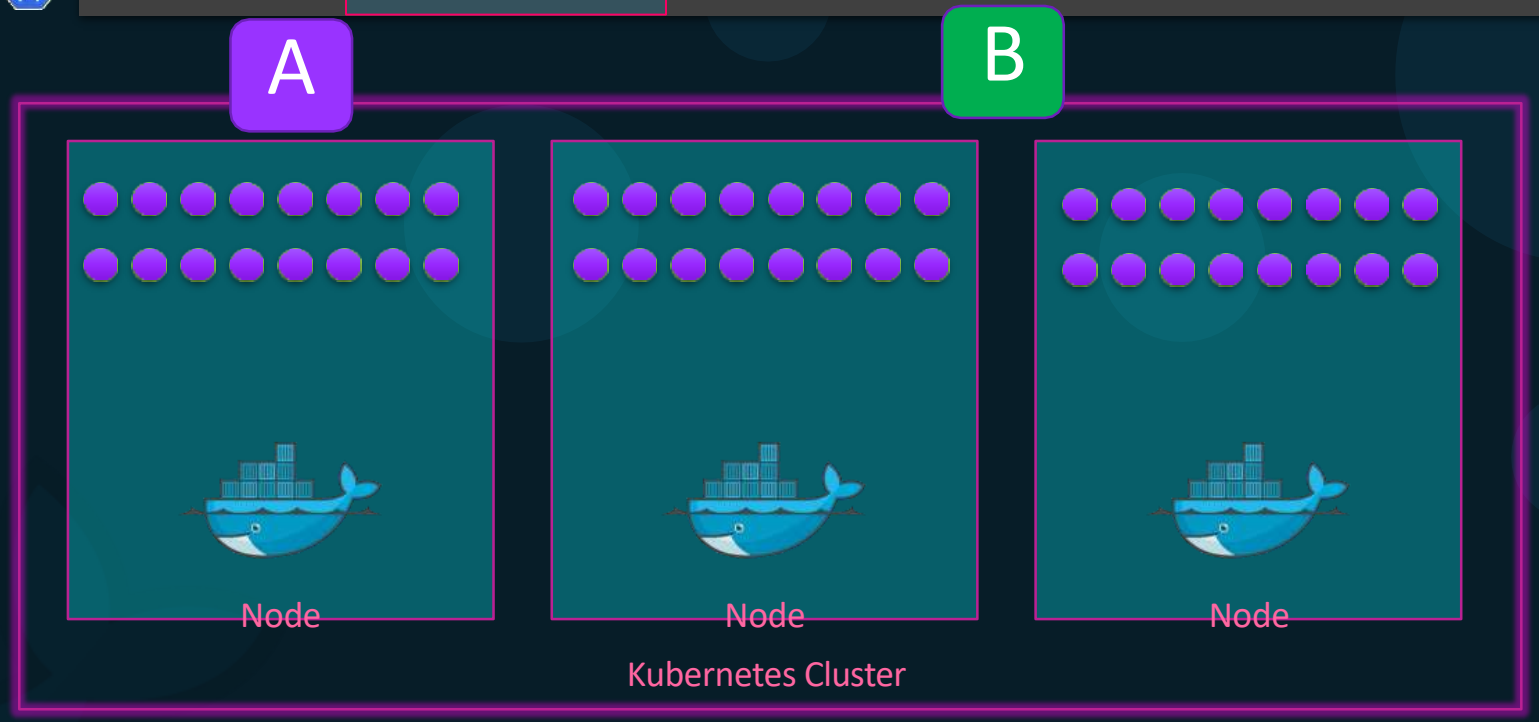
```
kubectl                    my-web-server --image=web-server:2
```

```
kubectl scale                    my-web-server
```

```
kubectl run --replicas=1000 my-web-server
```

```
docker run my-web-server
```





Security

Network

Storage

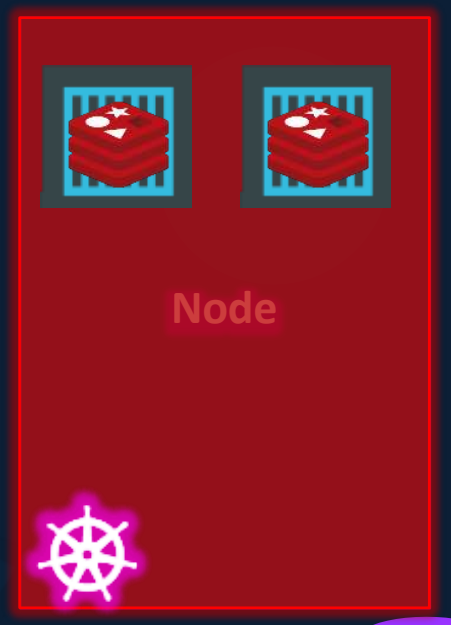
POD AutoScalers

Cluster AutoScalers

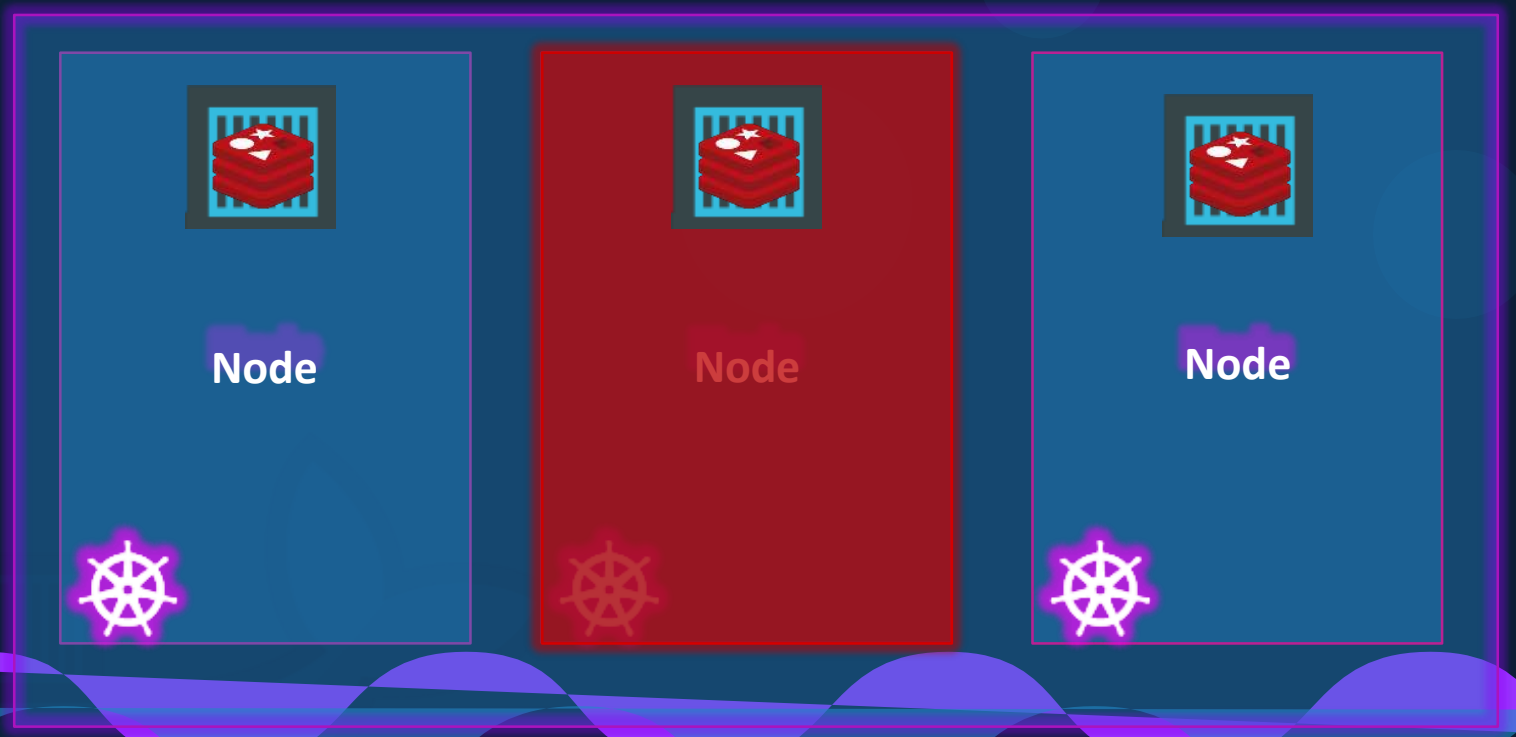
Amazon EKS

WISSEN

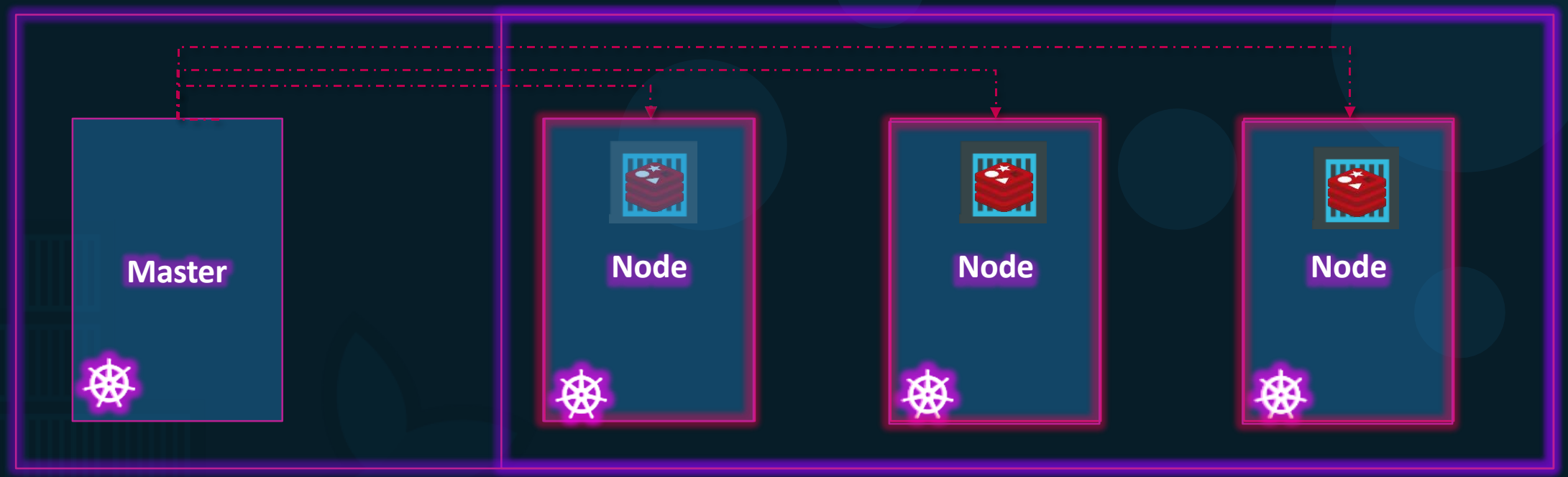
Nodes (Minions)



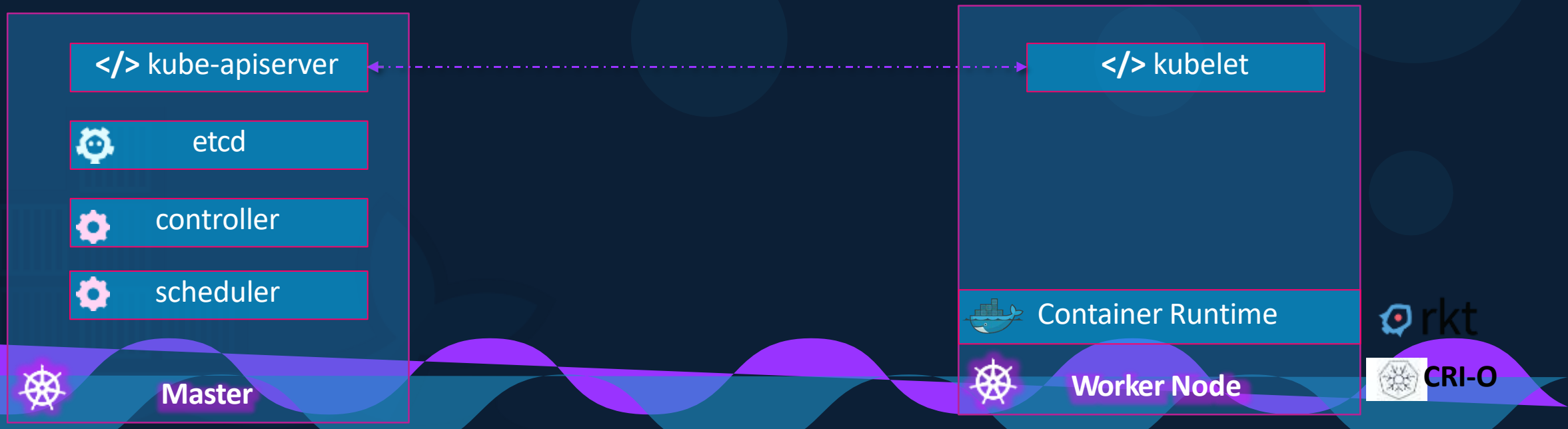
Cluster



Master



Master vs Worker Nodes



kubectl

```
kubectl run hello-minikube
```

```
kubectl cluster-info
```

```
kubectl get nodes
```

```
kubectl run my-web-app --image=my-web-app --replicas=100
```

CONCLUSION

WISSEN

THANK YOU