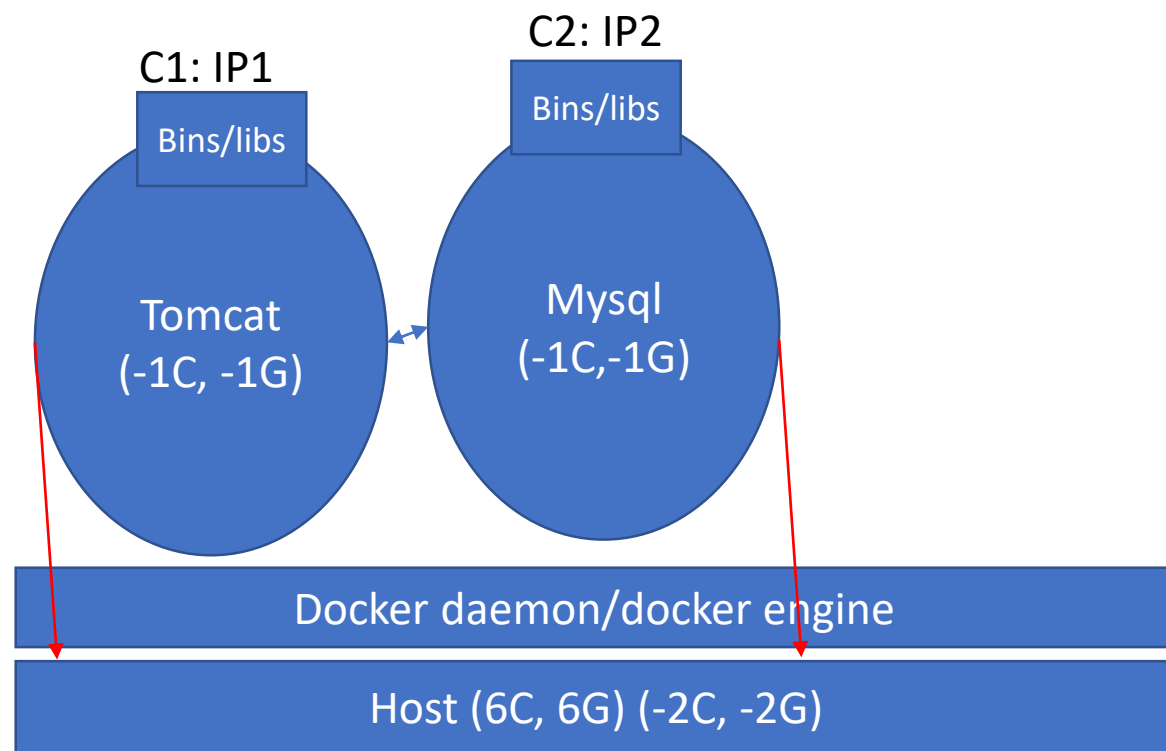1. speed: mins
2. Share-ability: snapshot/checkpoint/appliance: 1.6 GB
3. Repeat-ability: run sheets, scripts
4. Bang-for-buck: limited, fragmented
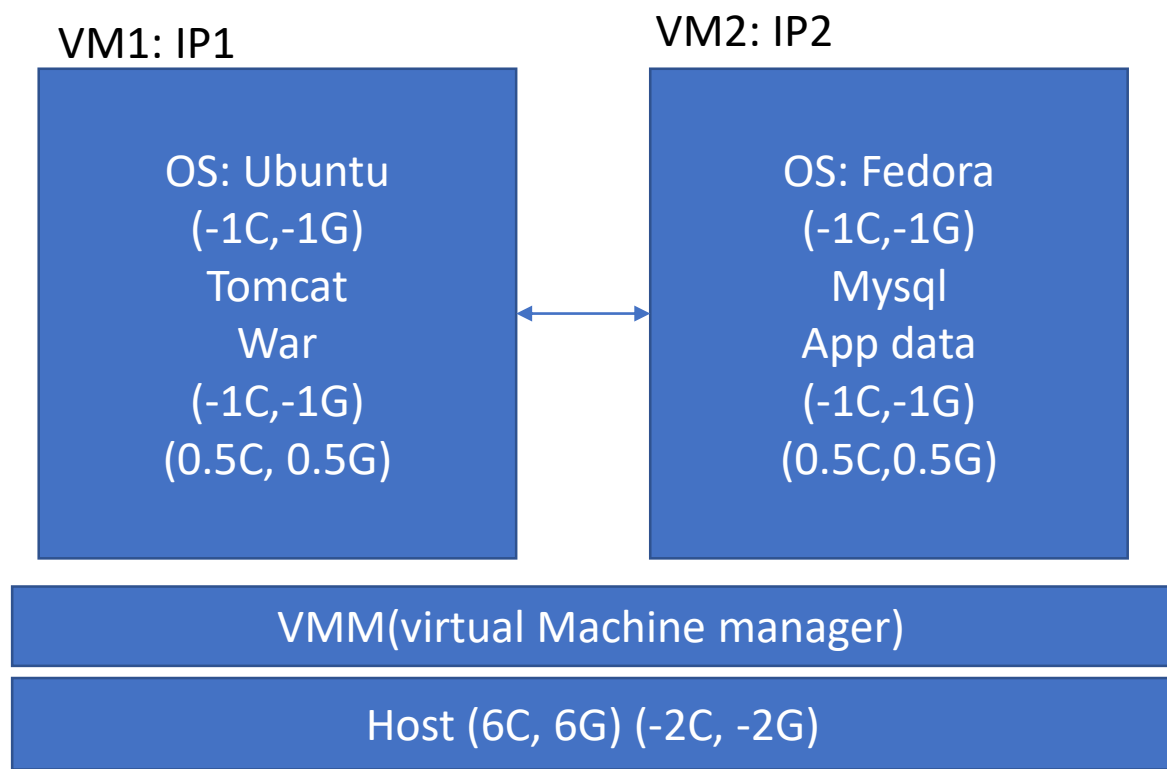
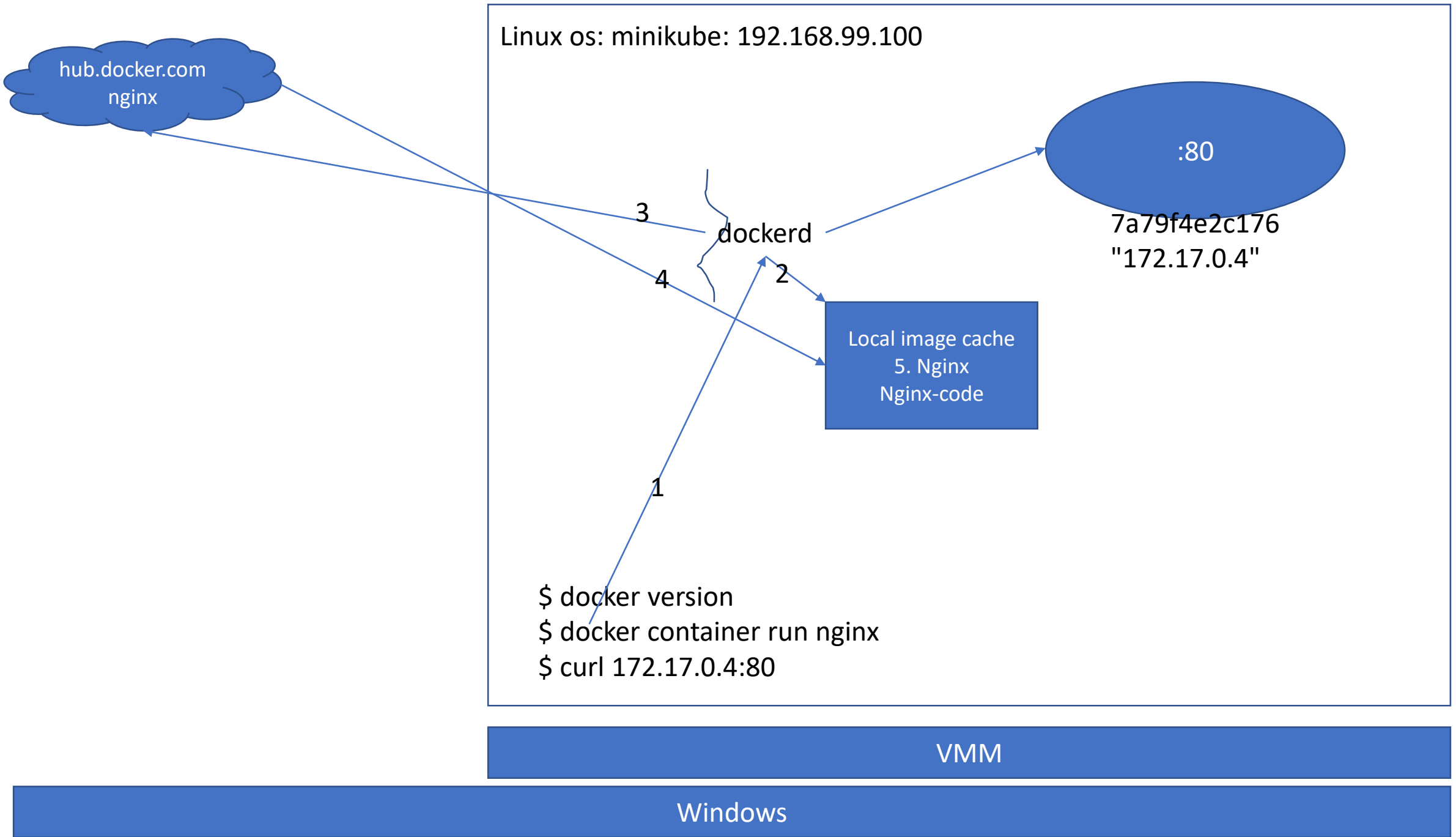1. speed: ms
2. Share-ability: images
3. Repeat-ability: DSL, code
4. Bang-for-buck: ++, not fragmented

VM1: IP1

OS: Ubuntu
(-1C,-1G)
Tomcat
War
(-1C,-1G)
(0.5C, 0.5G)

VM2: IP2

OS: Fedora
(-1C,-1G)
Mysql
App data
(-1C,-1G)
(0.5C,0.5G)

VMM(virtual Machine manager)

Host (6C, 6G) (-2C, -2G)

Isolation: Network, file system, process

C1: IP1

Bins/libs

Tomcat
(-1C, -1G)

C2: IP2

Bins/libs

Mysql
(-1C,-1G)

Docker daemon/docker engine

Host (6C, 6G) (-2C, -2G)

Isolation: Network, file system, process

Windows/host os

Minikube: 192.168.99.100

Virtualhostonly network
adapter #4
Low: 192.168.99.100
Upper: 192.168.99.255

Gateway:192.168.99.1

Docker0/bridge
Gateway: 172.17.0.1

Nginx:80

172.17.0.4:80

➢ Ping 192.168.99.100   -- will work

➢ Ping 172.17.0.4 -- cannot work

$ curl 172.17.0.4

Port-fwd/port-publish

http://192.168.99.100: 9999

9999

Image: Unified File System

shruthikeerthi/nginx-customized

nginx-code

7c6571209852

38e22b985a5f

nginx

ab56bba91343

Layer5 ~19MB	Layer4 ~19MB

Layer3 ~22B (soft link)

Layer2 ~57MB (installed nginx)

debian

Layer1~70MB

nginx

nginx-code

Containers fS : ephimeral

minikube

app
-- /var/log/access.log
/var/log/error.log

/var/lib/docker/volumes/5c64f44a8e33ca0f49a82c044
e9db87d8e568413744d90605ca8e30cb9cbab9c/_data

/var/lib/docker/volumes/5aef8ea1e6723a12ecb353
b6b923de933051a68516b096e5035a3fa01bb0e18d
/_data – 100 users

Mysql
/var/lib/mysql

/var/lib/docker/volumes/projdb/_data

1. ipaddress→ dns service
2. Better isolation: w1 -> db , w2 -> db, w1 – X - w2

minikube

docker0   Bridge: 172.17.0.0/16

W2-db-net: 172.19.0.0/16

w2

172.19.0.2

X

w1-db-net: 172.18.0.0/16

db

w1

172.18.0.3
172.19.0.3

172.18.0.2

minikube

9090

ghost
Host=blog-db
Port:3306
User=root
Pwd=welcome

Mysql:5.7
3306

Blog-data

blog-db: 172.18.0.4

172.18.0.6
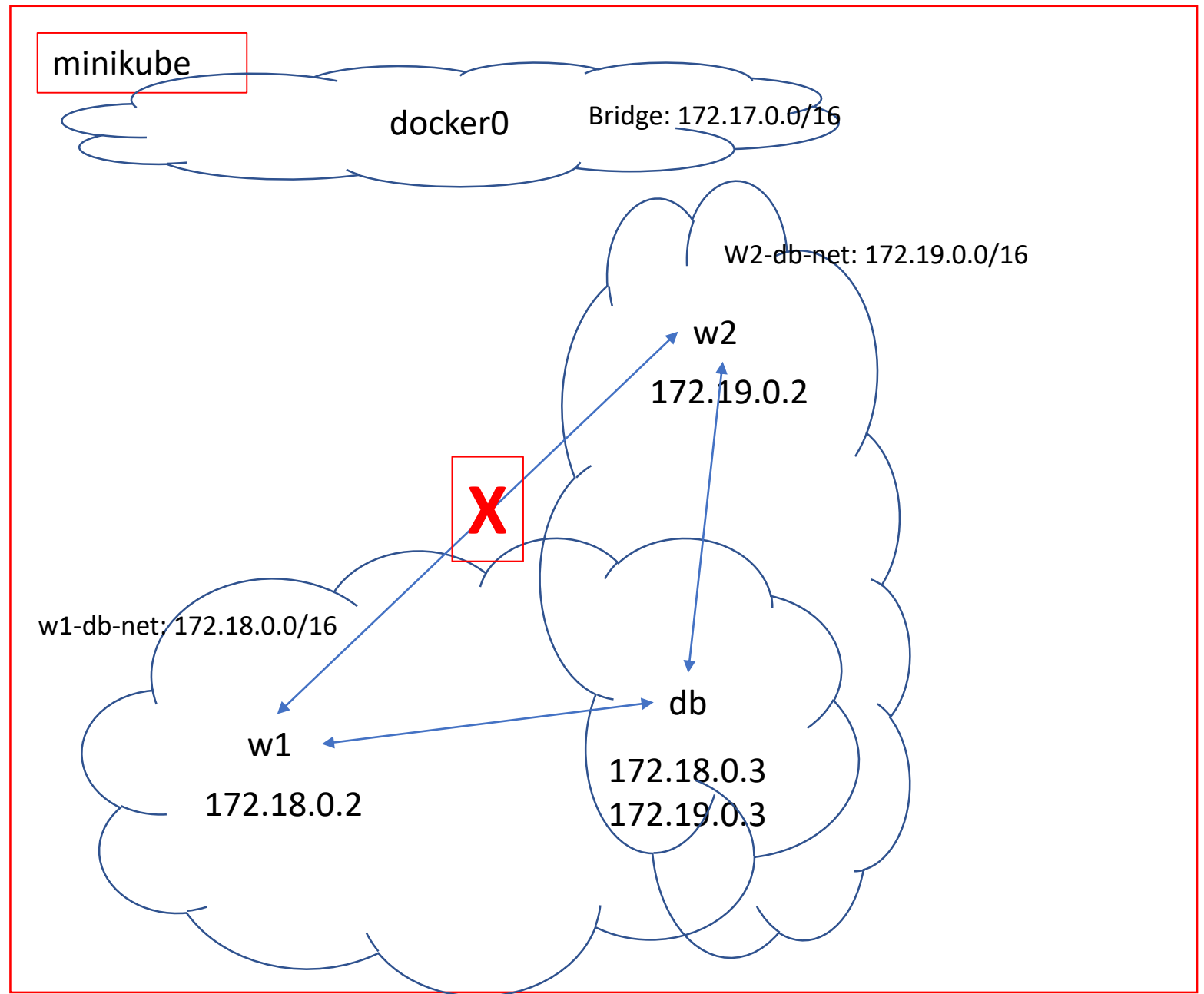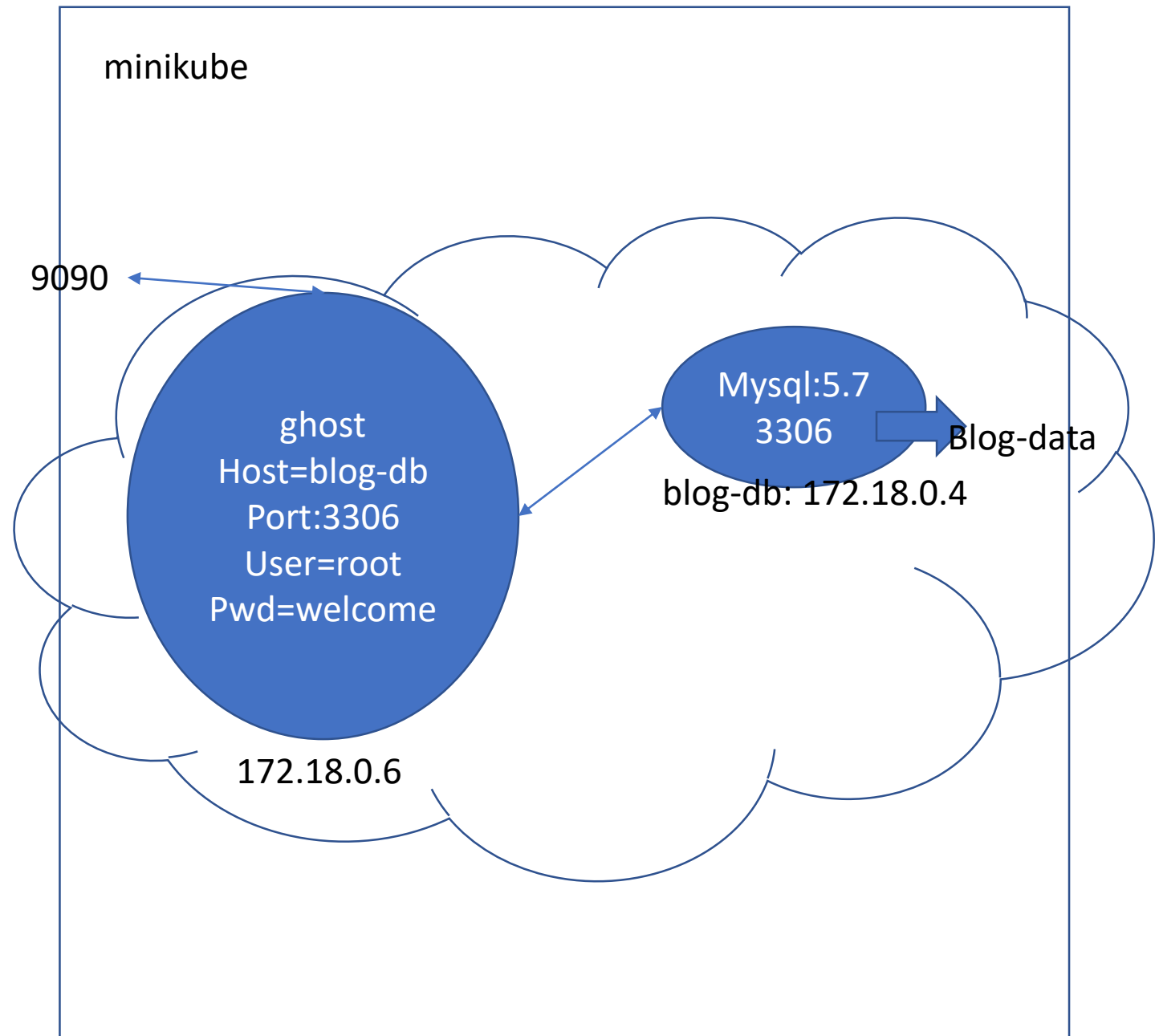
Container Orchestration

1. Application scaling
2. Load balancing
3. Interact and coordinate between multiple dockerd
4. Networking
   1. Ip address management
   2. No node awareness
5. Dns services
6. Node (VM, on prem,on cloud, physical) management -> cluster management
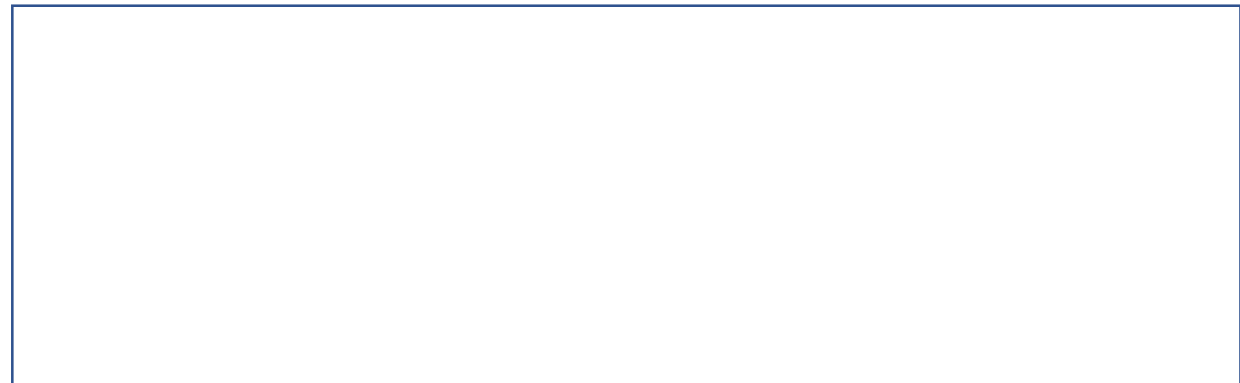
Products:
K8s
Docker swarm

Node1 8c, 124g

ghost

db

dockerd

172.20.0.2

Node2 8c, 124g

ghost

ghost

dockerd

172.21.0.5
Host: db ip:2345
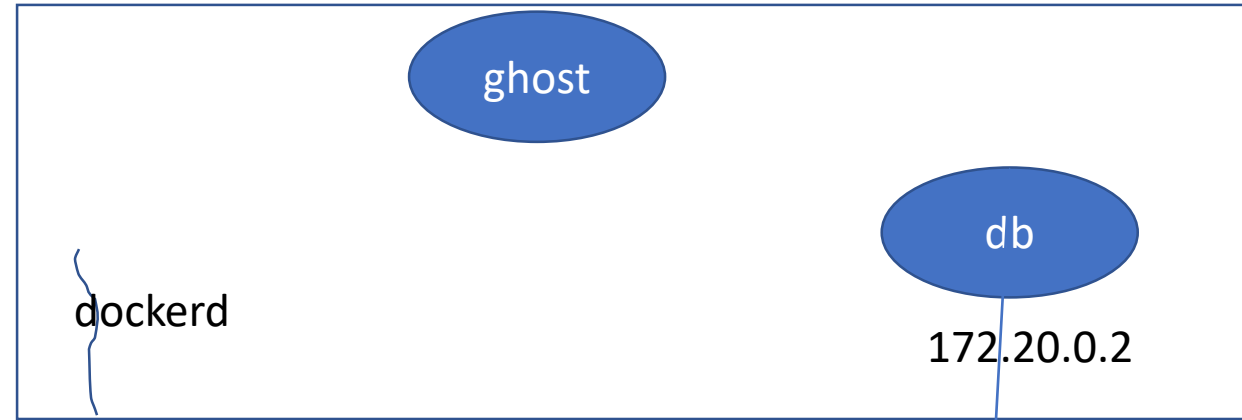
Node3 8c, 124g

Docker Swarm

1. Open source, Docker
2. Native support for docker containers
3. Learning curve is smooth
4. DS → docker
5. Vendor-lockin
6. Fairly new

K8s

1. Open source, Google
2. Terminology or adapters
3. Learning curve is a litter steeper
4. K8s -> any container provider which implement Open Container Spec
5. No vendor lockin
6. Battle tested → benchmarked against 10000's nodes

K8s language

adapter

dockerd

nginx

Docker – K8s
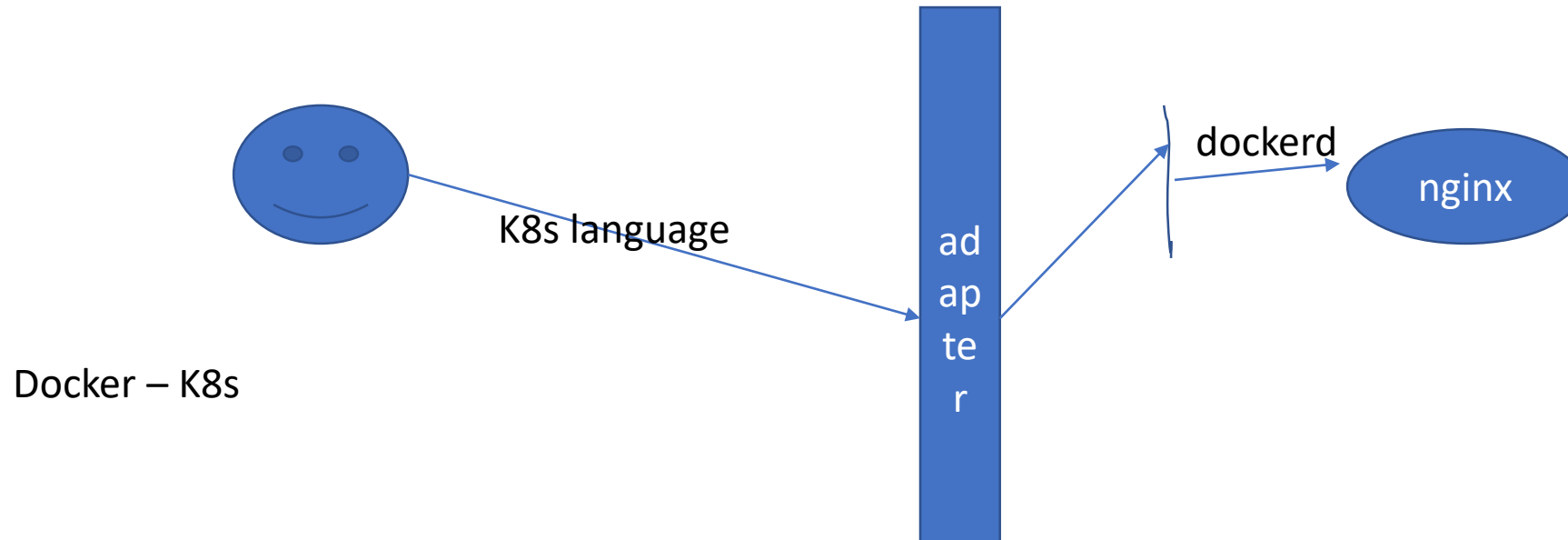
Learning K8s
1. Terms and concepts k8s – single node cluster
2. Communication – single node clust
3. Architecture
4. Multi-node cluster k8s