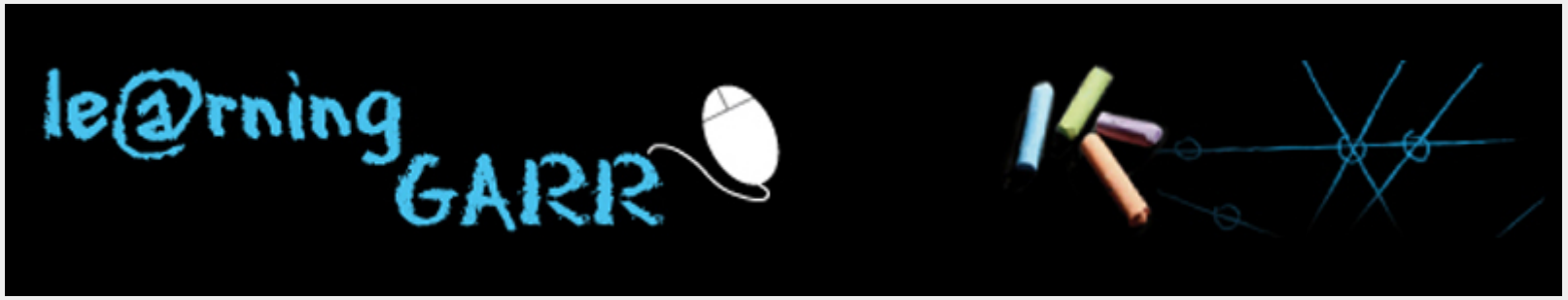
DOCKER VS VIRTUALIZZAZIONI
28 MAGGIO 2018
ROMA

RIGRAZIAMENTI



ABOUT ME

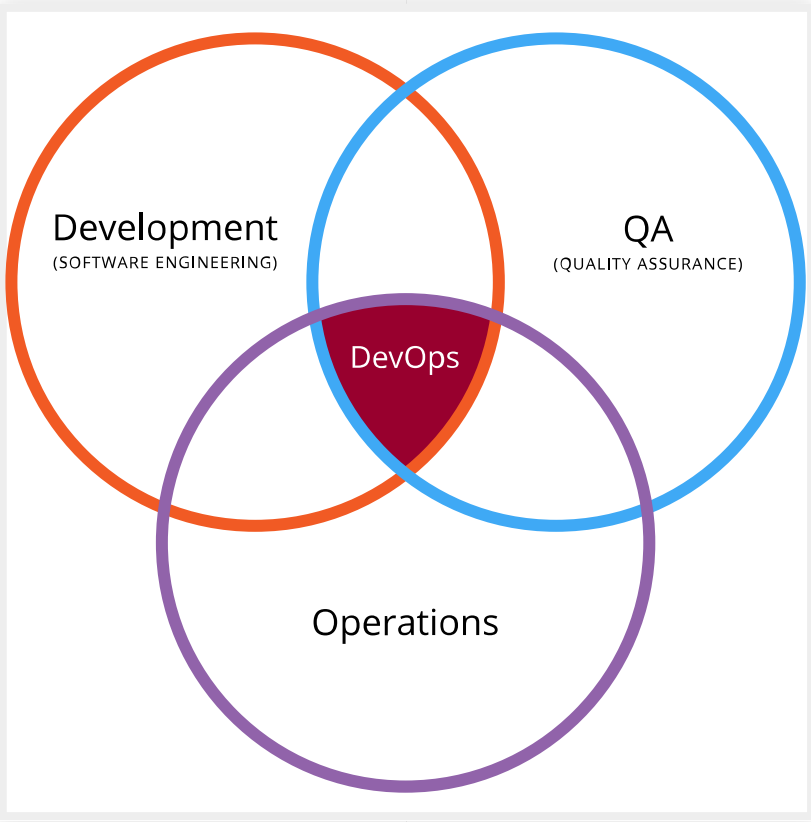
- ITPro > 25 year
- Senior Software Developer at INNOTEAM s.r.l.
- Linux Foundation Member
- CTS Senior Member at Federation IDEM AAI
- ICT Administrator at Università Politecnica delle Marche
- Istruttore Informatico at Comune di Chiaravalle (AN)
- Member of WindowServer.it community
- Member of DevMarche community

I.I.I.D.

INFRASTRUTTURA, ISOLAMENTO, IDEMPOTENZA E DEVOPS

I VANTAGGI E I VINCOLI DELLA DEFINIZIONE FORMALE DI UN'INFRASTRUTTURA

DEVOPS



Una completa automazione assicura la facile ripetibilità dei rilasci e riduce gli errori nell'operazione.

INFRASTRUTTURA

Insieme di materiale ed elementi singoli che collaborano a sostenere, produrre ed erogare beni e servizi. Nel mondo ICT sono *hardware* e/o *artefatti software* che collaborano per risolvere problemi complessi.

ISOLAMENTO

Il concetto di *Isolamento* dipende dalla definizione di due domini che chiameremo per convenzione *Interno* ed *Esterno* e dalla definizione di una superficie di separazione tra Interno ed Esterno chiamata per convenzione *Confine*. L'*Isolamento* si applica sul *Confine* e condiziona lo scambio tra *Interno* ed *Esterno*.

IDEMPOTENZA

È la proprietà di un’operazione che applicata una o più volte non cambia il risultato prodotto.

L’applicazione multipla ed asincrona della funzione con questa proprietà non introduce collisioni o comportamenti non deterministici.

STATO

Contenuto informativo temporaneo o persistente nell'artefatto che annulli la proprietà di *idempotenza*. Per ripristinare la proprietà di *idempotenza*, l'artefatto delega ad elementi esterni la gestione dei propri stati.

DEFINIZIONE FORMARE D'INFRASTRUTTURA

Descrizione oggettiva dell'*infrastruttura* la cui implementazione viene delegata ad una procedura automatica.

ESEMPIO DI DEFINIZIONE FORMALE

```
version: '2'

services:
  web:
    build: ./php/
    volumes:
      - /var/www:/var/www/html:rw
    ports:
      - "80:80"
      - "443:443"
    links:
      - mysql
    environment:
      XDEBUG_CONFIG: remote_host=10.211.55.2

  mysql:
```

VANTAGGI

- Resilienza
- Scalabilità Dinamica
- Documentazione
- Replicabilità
- Diaccoppiamento dall’Hardware
- Test d’Infrastruttura automatizzabili
- Condivisione

VINCOLI

- Esatta conoscenza dei domini che concorrono nell'Infrastruttura
- Oneroso per aggiustamenti di lieve entità
- Rigore nel Workflow di lavoro
- Studio, Analisi, Test, Tempo


QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation

DOCKER

UN FACILITATORE PER UTILIZZARE I CONTAINER E LE TECNOLOGIE AD ESSI COLLEGATE

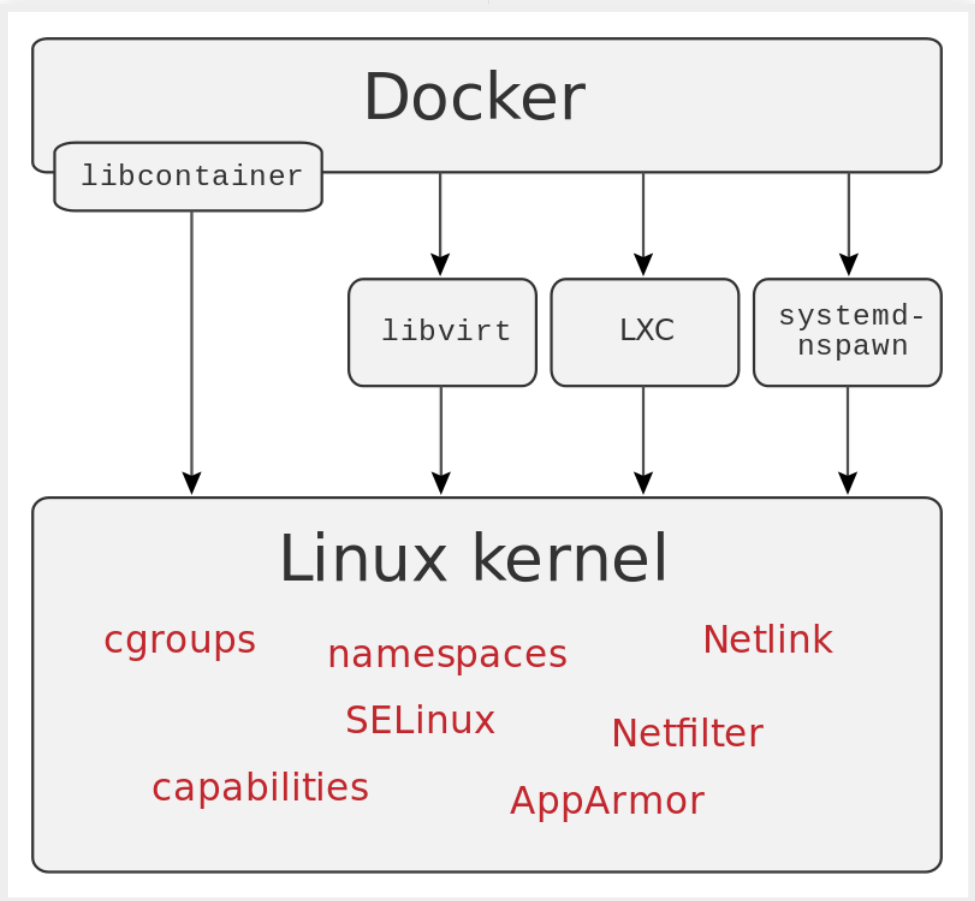
COSA È DOCKER

UN FACILITATORE?

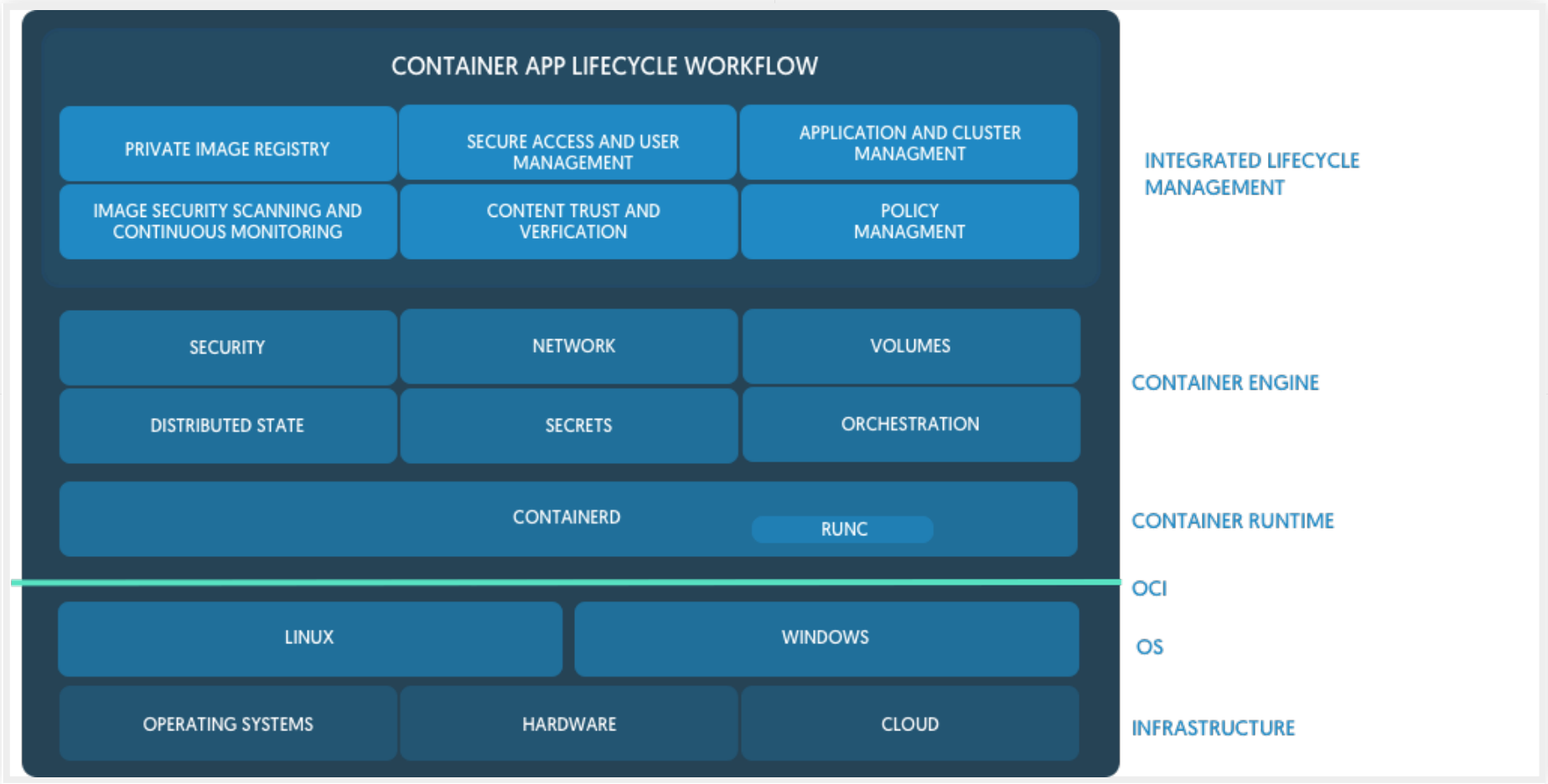
UN HUB DI TECNOLOGIE?

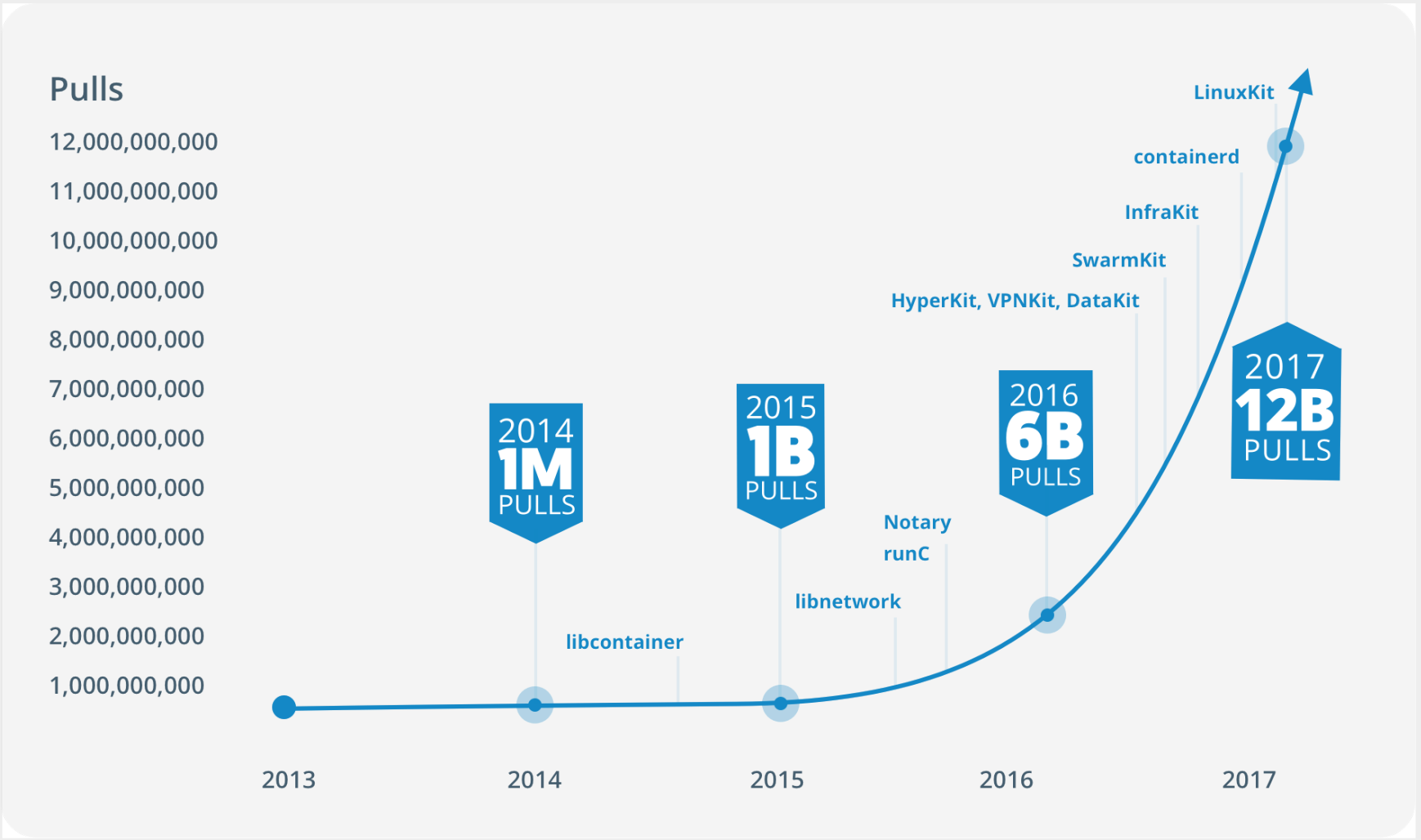
UN SIGNIFICATO NUOVO ALLA PAROLA *INFRASTRUTTURA*
E UN MODO NUOVO PER COSTRUIRLA.

DOCKER, GLI INIZI

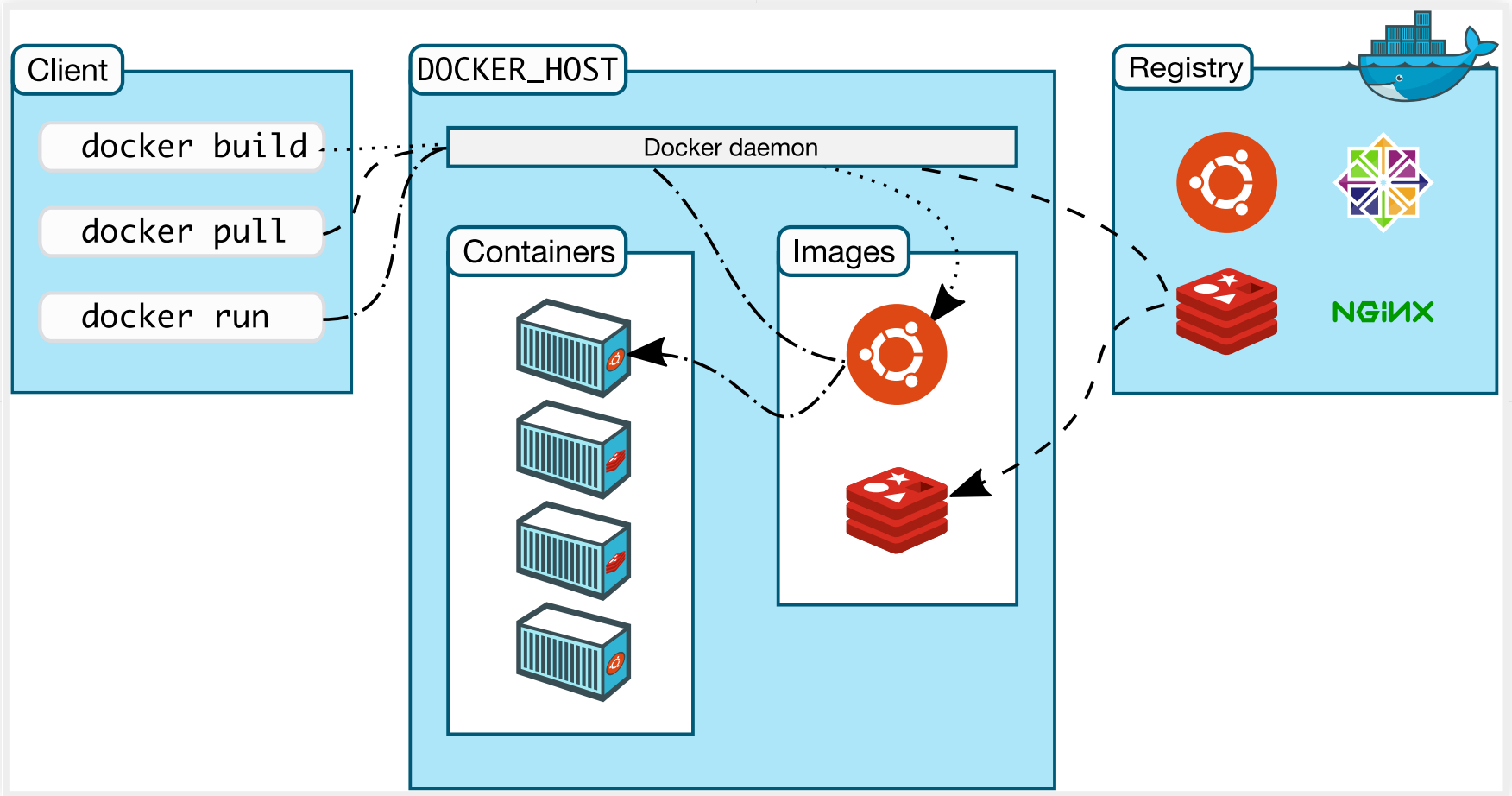


DOCKER OGGI


CONTAINER APP LIFECYCLE WORKFLOW
PRIVATE IMAGE REGISTRY
SECURE ACCESS AND USER MANAGEMENT
APPLICATION AND CLUSTER MANAGMENT
IMAGE SECURITY SCANNING AND CONTINUOUS MONITORING
CONTENT TRUST AND VERFICATION
POLICY MANAGMENT
INTEGRATED LIFECYCLE MANAGEMENT
SECURITY
NETWORK
VOLUMES
DISTRIBUTED STATE
SECRETS
ORCHESTRATION
CONTAINER ENGINE
CONTAINERD
RUNC
CONTAINER RUNTIME
OCI
LINUX
WINDOWS
OS
OPERATING SYSTEMS
HARDWARE
CLOUD
INFRASTRUCTURE


Pulls
12,000,000,000
11,000,000,000
10,000,000,000
9,000,000,000
8,000,000,000
7,000,000,000
6,000,000,000
5,000,000,000
4,000,000,000
3,000,000,000
2,000,000,000
1,000,000,000
2013
2014
2015
2016
2017
2014 1M PULLS
2015 1B PULLS
2016 6B PULLS
2017 12B PULLS
libcontainer
libnetwork
Notary
runC
HyperKit, VPNKit, DataKit
SwarmKit
InfraKit
containerd
LinuxKit

CONTAINERS WORKFLOW

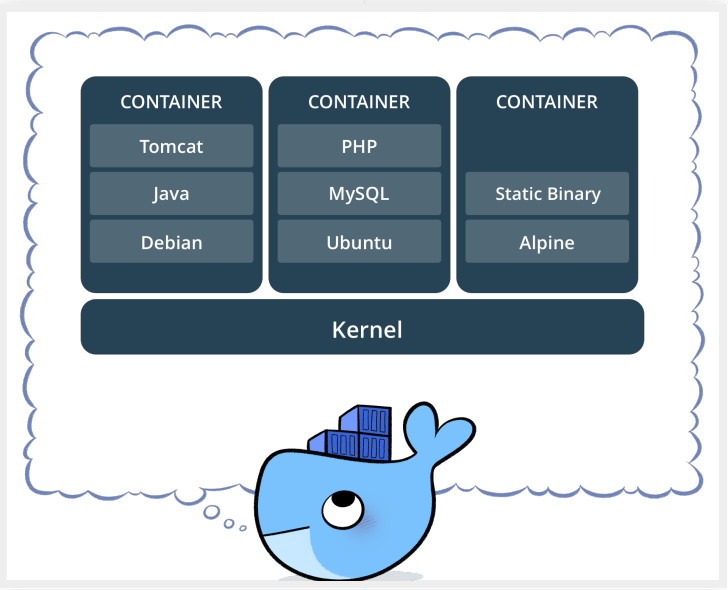


L'IMMAGINE SECONDO DOCKER.INC

> *A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings.*
>
> — *Official Docker website*

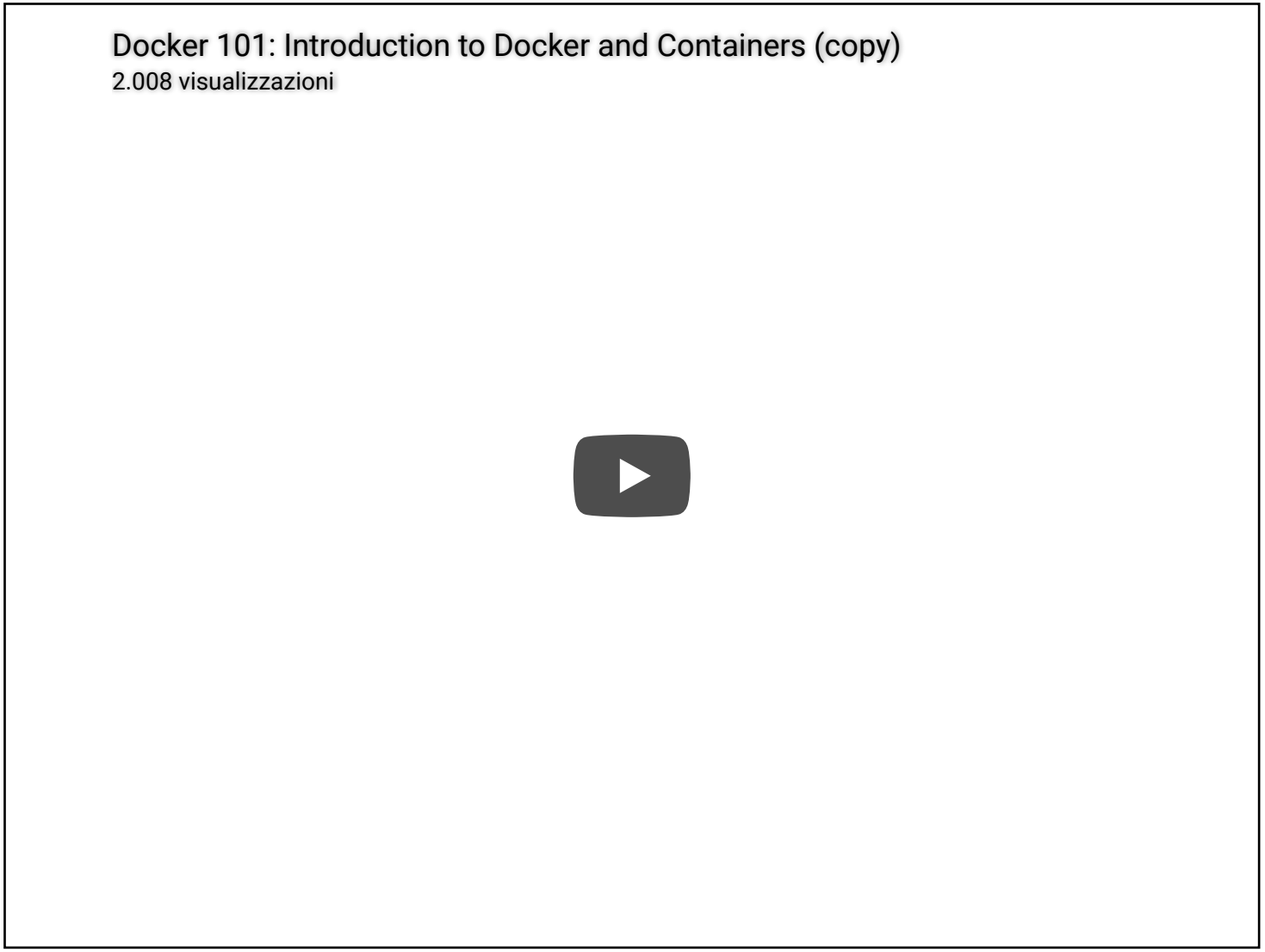
È un ***artefatto***. Il *Container* è la sua rappresentazione operativa come il processo lo è del codice.

IL CONTAINER



Il *Container* è un elemento isolato e autosufficiente, accessibile attraverso la rete. Esso astrae a livello applicativo le dipendenze necessarie al codice.

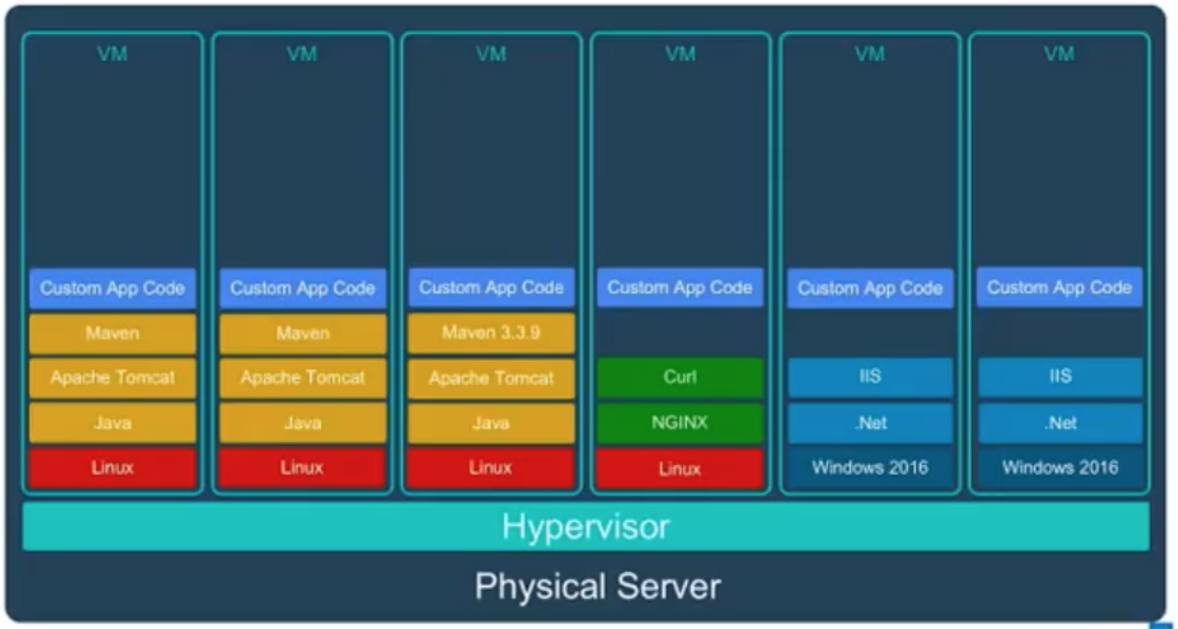
DOCKER 101: INTRODUCTION TO DOCKER



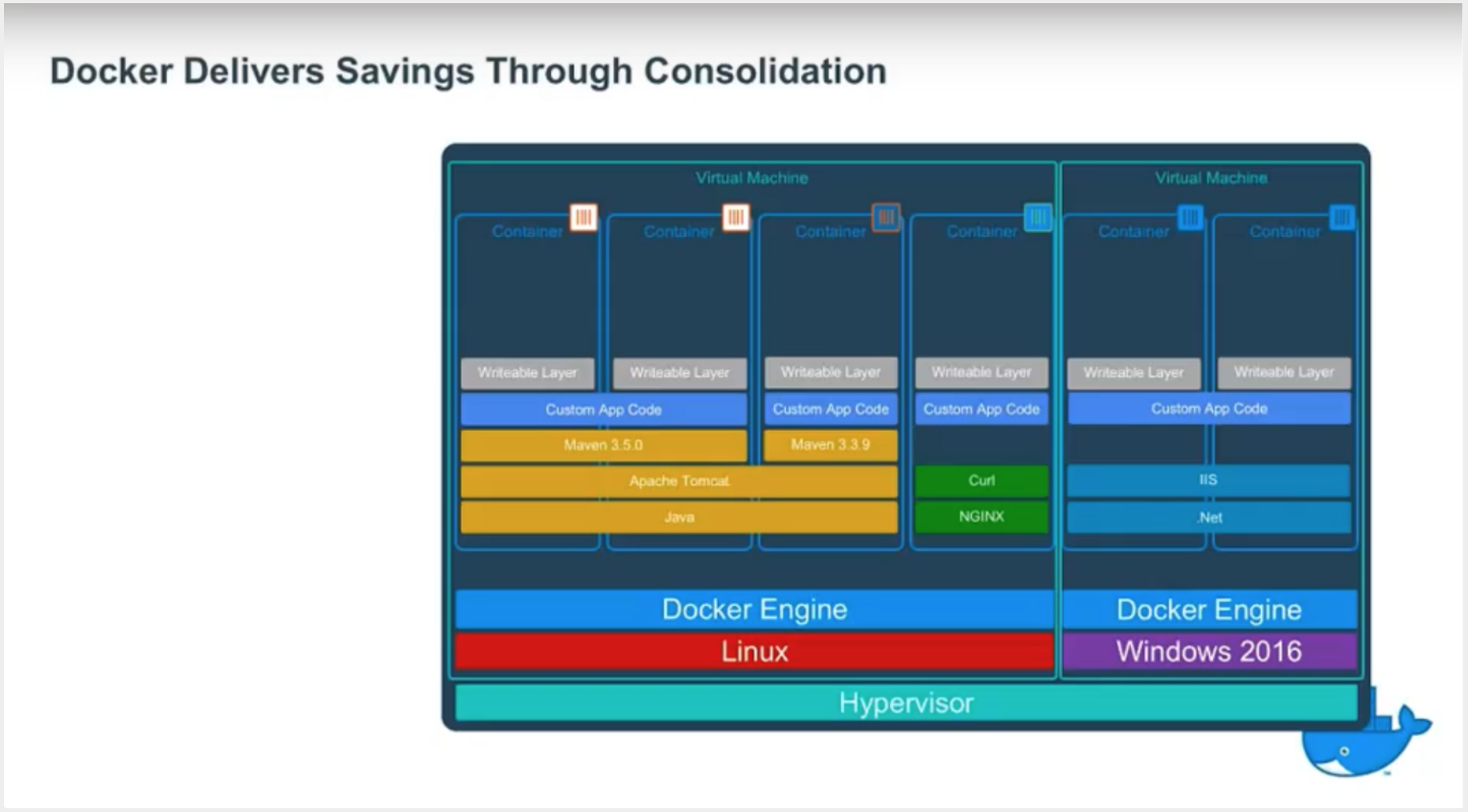
VM CONSOLIDATION

Our Current State: Virtual Machines

- One app per VM
- Separate OS copy
- Separate app framework
- ”Black box” to hypervisor
- App is tracked elsewhere



CONTAINER CONSOLIDATION



CREAZIONE DI UN CONTAINER

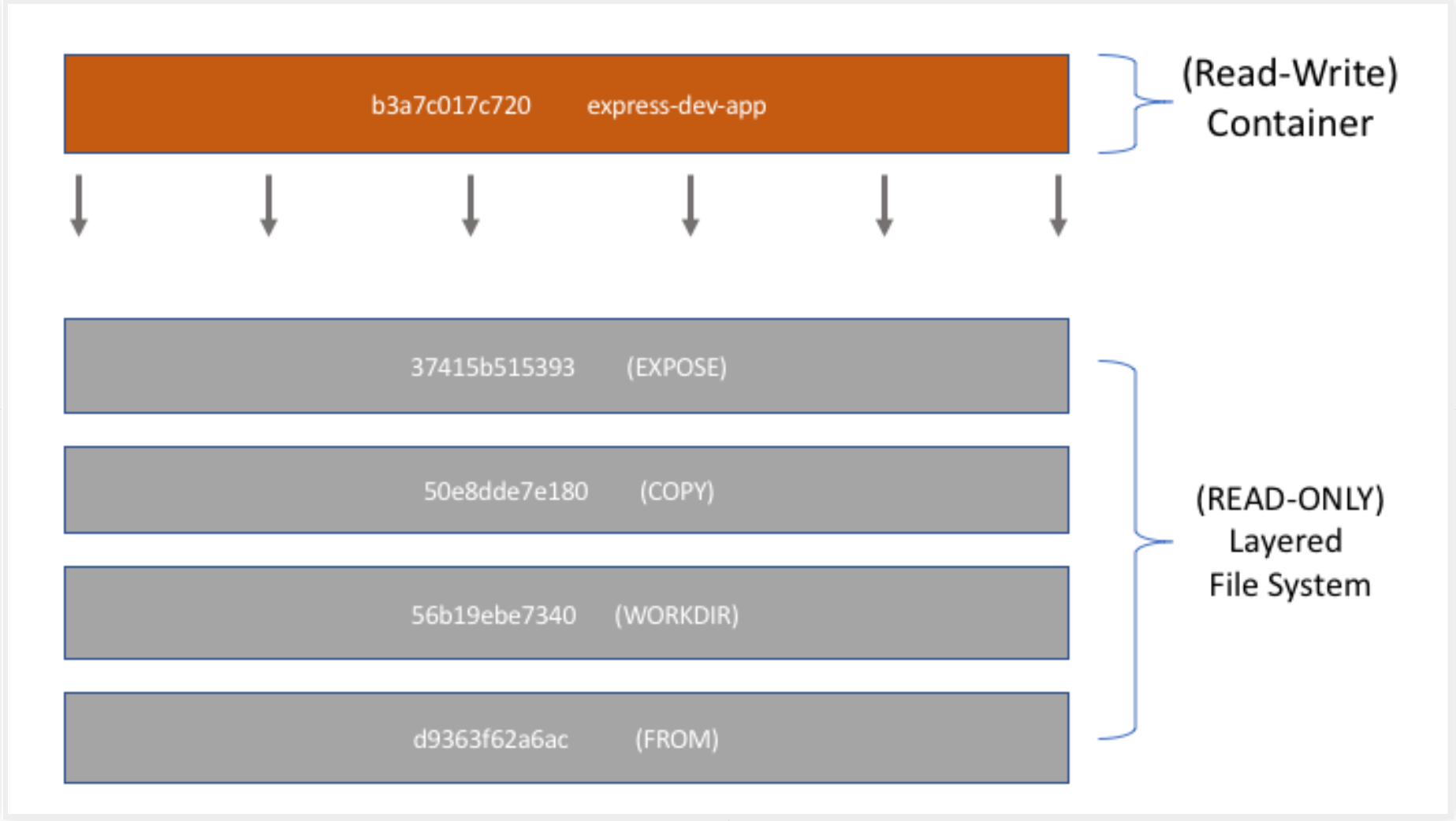
```
bash-4.4$ docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
1be7f2b886e8: Pull complete
6fbc4a21b806: Pull complete
c71a6f8e1378: Pull complete
4be3072e5a37: Pull complete
06c6d2f59700: Pull complete
Digest: sha256:e27e9d7f7f28d67aa9e2d7540bdc2b33254b452ee8e60f3888
Status: Downloaded newer image for ubuntu:latest
root@66170252cf62:/#uname -a
Linux 66170252cf62 4.9.75-linuxkit-aufs #1 SMP Tue Jan 9 10:58:17
```

CREAZIONE DI UN'IMMAGINE

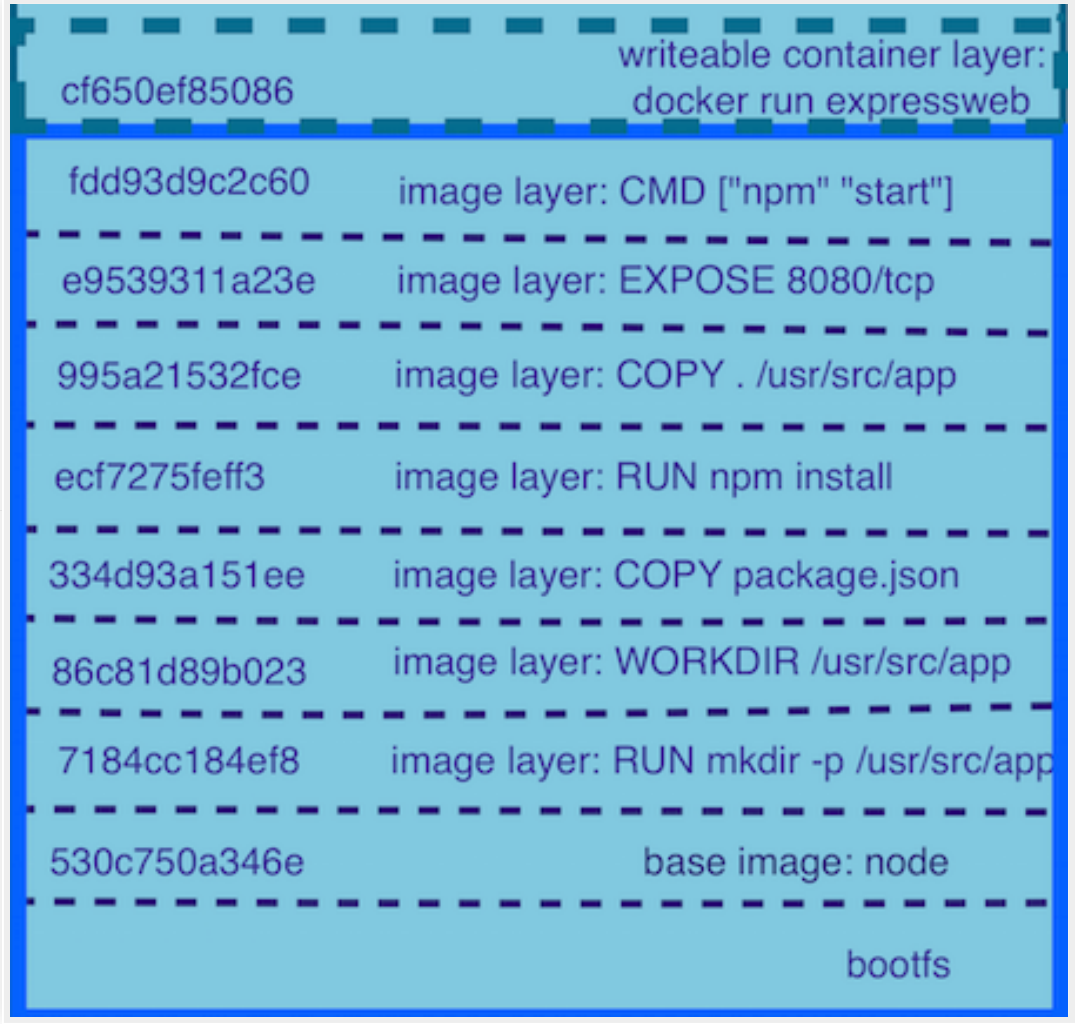
```
FROM php:5.6-apache
# Install XDebug
RUN yes | pecl install xdebug \
    && echo "zend_extension=$(find /usr/local/lib/php/extensions/
    && echo "xdebug.remote_enable=on" >> /usr/local/etc/php/conf.
    && echo "xdebug.remote_autostart=off" >> /usr/local/etc/php/c

# Install PHP exrensions.
RUN apt-get update \
        && apt-get install -y \
            imagemagick \
            graphicsmagick \
            zip \
            unzip \
            wget \
            curl \
```

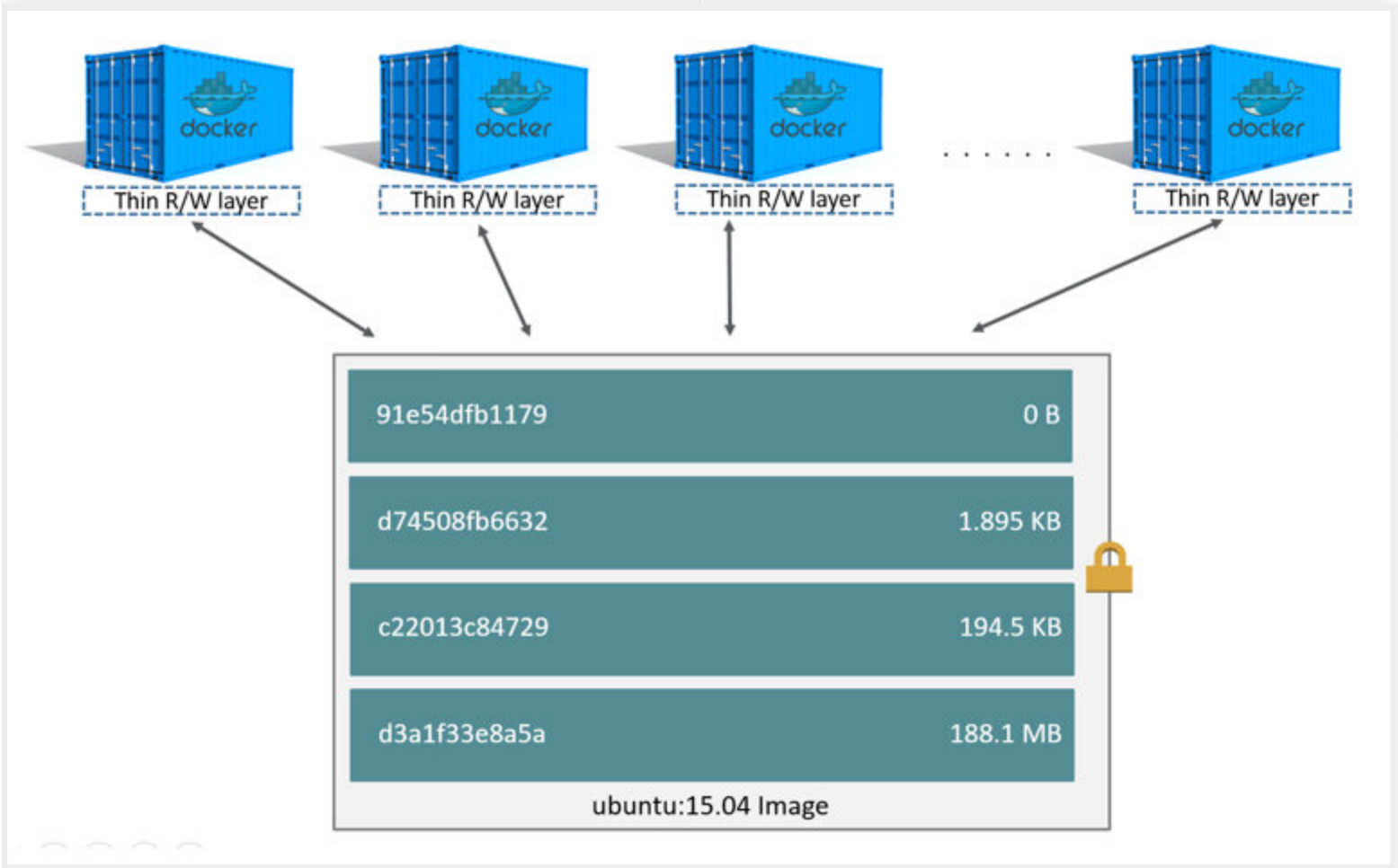
From: ./02_Docker/code/01/dockerfile


b3a7c017c720 express-dev-app
(Read-Write)
Container
37415b515393 (EXPOSE)
50e8dde7e180 (COPY)
56b19ebe7340 (WORKDIR)
d9363f62a6ac (FROM)
(READ-ONLY)
Layered
File System

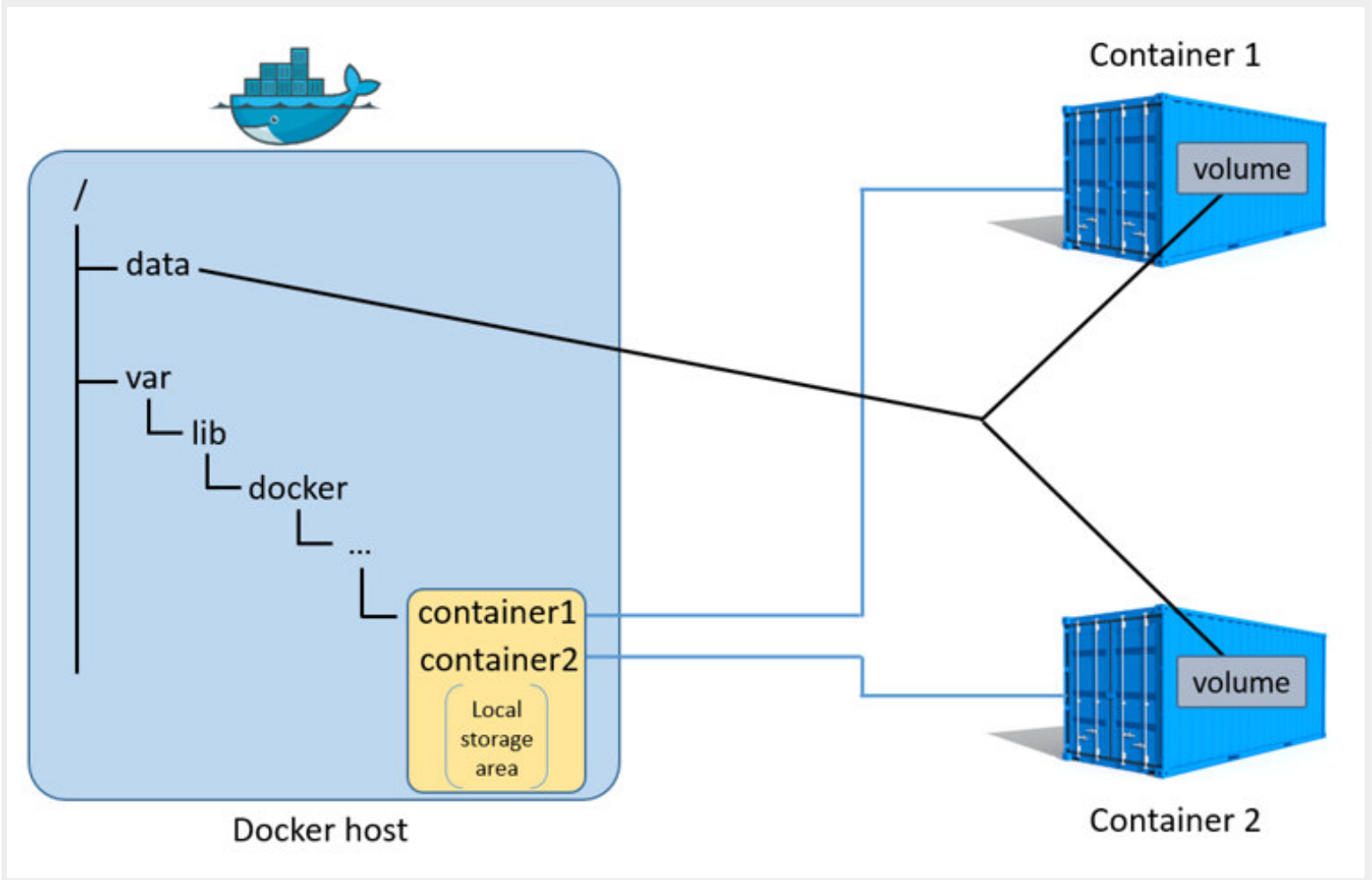
COSA SUCCEDE DOPO UN *DOCKER RUN*



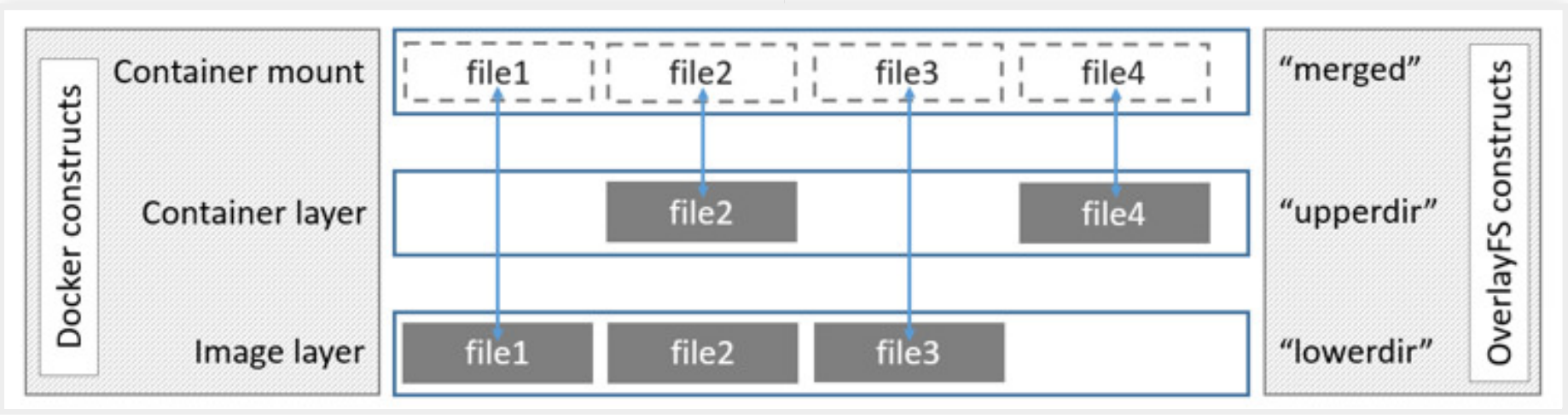
LAYERS CONDIVISI



HOST-FILESYSTEM CONDIVISI



FILES VISION IN OVERLAYFS



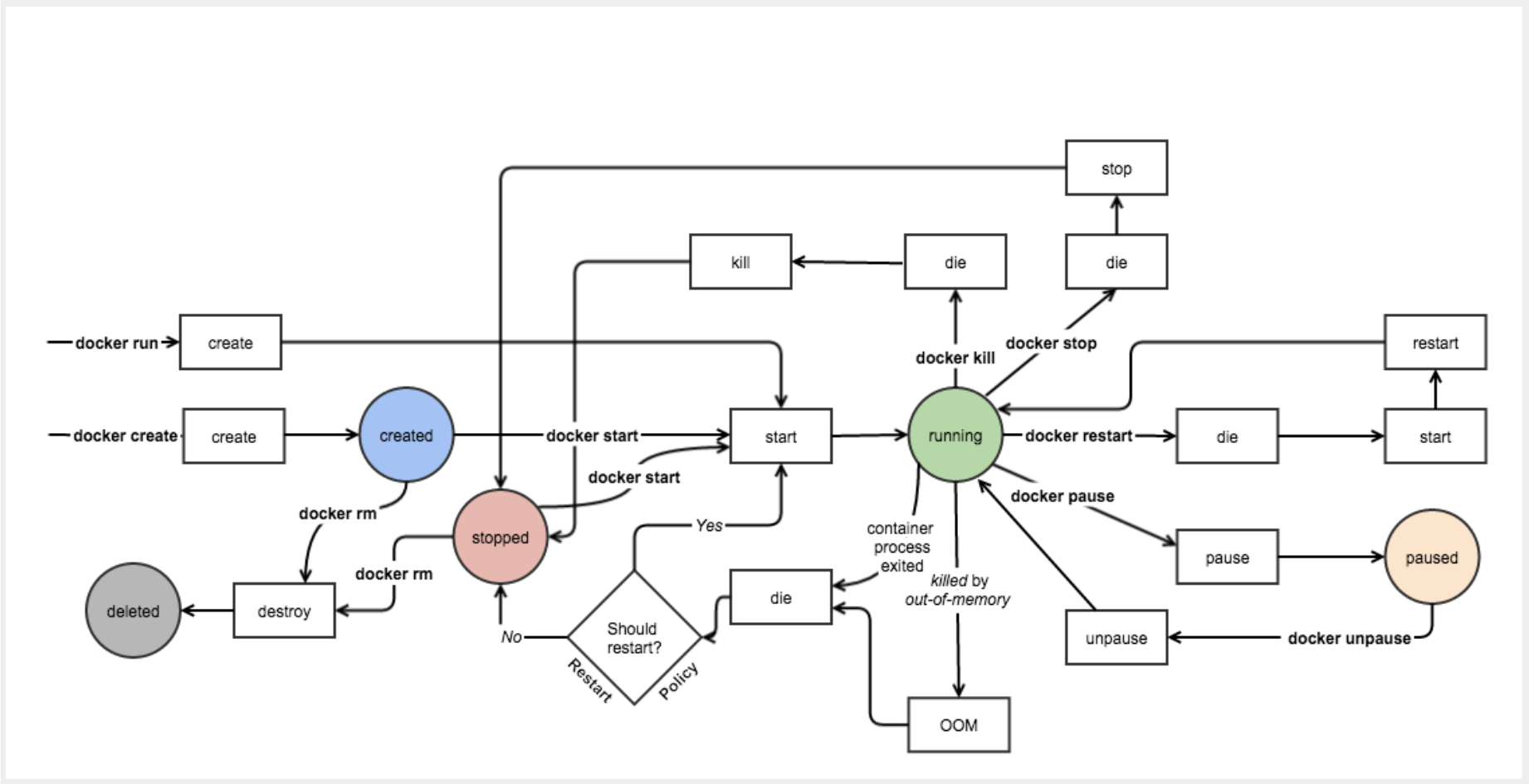
GLOSSARIO DOCKER BASE

- *Docker Engine*: deamon gestore dei elementi di un *Docker Host*
- *Docker Client*: CommandLine Interface per interagire con la *Docker Engine* locale o remota
- *Docker Image*: readonly template generato da `docker build` da cui avviare uno o più Container
- *Docker Container*: artefatto attivo prodotto da `docker run` a partire da un'immagine

GLOSSARIO DOCKER AVANZATO

- *Docker Host*: Sistema fisico o virtuale in cui viene eseguita la *Docker Engine*
- *Docker Registry*: Repository di immagini Pubblico hub.docker.com o Privato
- *Docker Compose*: Orchestrator di infrastrutture multicontainer organizzati in servizi
- *Docker Swarm*: Cluster multihost ad orchestrazione automatica

WORKFLOW COMANDI DOCKER



- Build Image (Interpretazione del Dockerfile)

```
docker build dockerID/imagename:tag .
```

- Run Container (creazione del Container da un'immagine)

```
docker run -d dockerID/imagename:tag
```

- Pull Image (scarico di un'immagine)

```
docker pull dockerID/imagename:tag
```

- Push Image (invio di un immagine al repository)

```
docker push dockerID/imagename:tag
```

ANALISI DI UN DOCKERFILE

```
# vim:set ft=dockerfile:
FROM debian:jessie

# add our user and group first to make sure their IDs get assigne
RUN groupadd -r mysql && useradd -r -g mysql mysql

# add gosu for easy step-down from root
ENV GOSU_VERSION 1.10
RUN set -ex; \
    \
    fetchDeps=' \
        ca-certificates \
        wget \
    '; \
    apt-get update; \
```

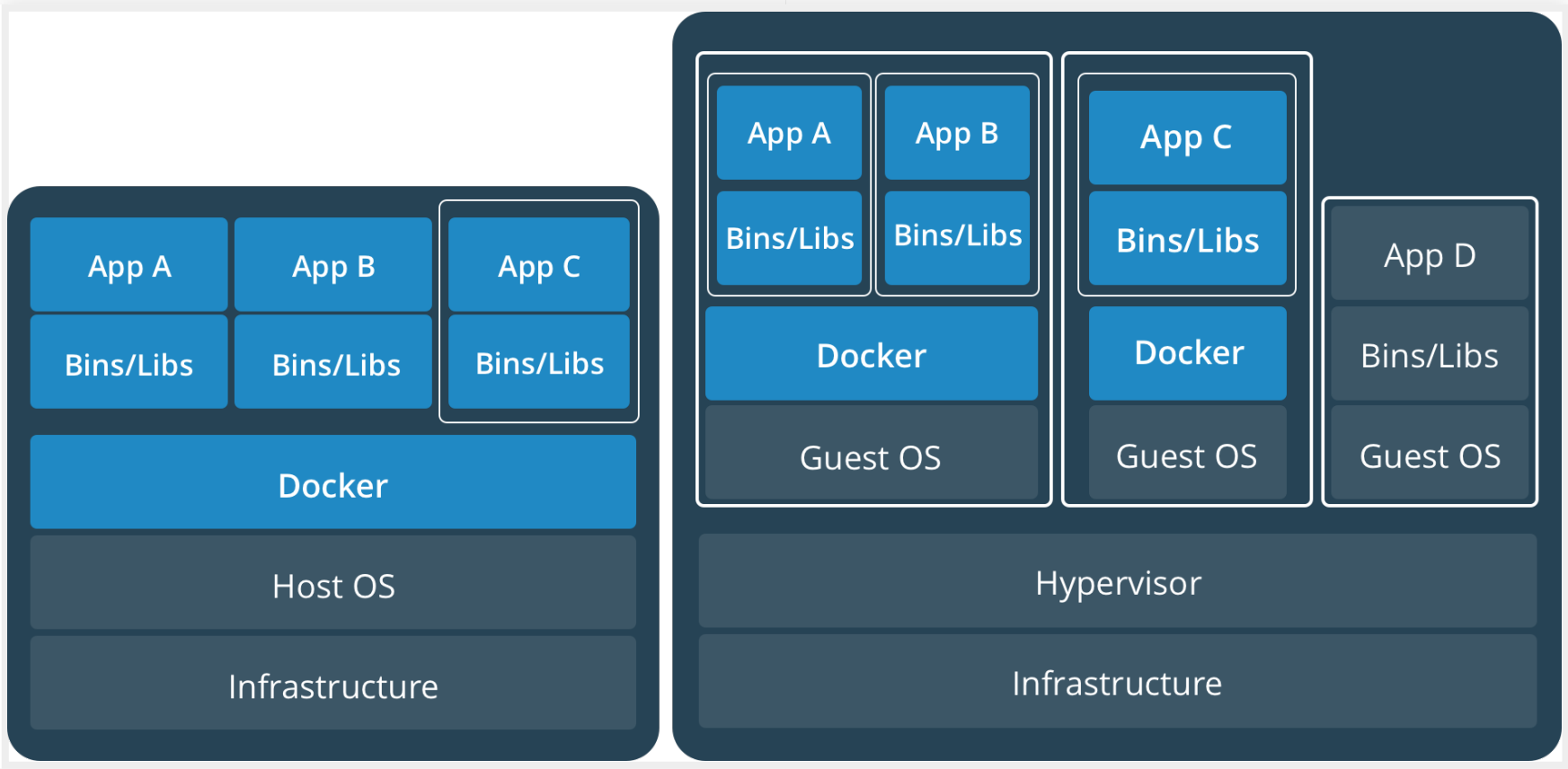
From: ./02_Docker/code/02/dockerfile


QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation

CONTAINER VS HYPERVISOR

LE DUE FACCE DELLA STESSA MEDAGLIA: PRO, CONTRO ED ESEMPI D'USO

VM E CONTAINER A CONFRONTO



QUANDO USO LE VM

- Hardware da simulare
  Es. *MAME is a multi-purpose emulation framework.*
- Software monolitico e fortemente interconnesso
  Es. *GUI unita alla Business logic*
- Software GUI e Console esclusi quelli Web-based
- Software non scalabile
- Quando lo sforzo di Automazione delle Ops produce un ROI negativo

QUANDO USO I CONTAINER

- Bounderies dei domini ben definiti
- Software cli-based su OS Linux o Windows
- Software su architettura Client/Server connessi attraverso Network TCP/IP
- Alta scalabilità e automazione dei processi Op
- Assenza di stato nei Container Applicativi, gli stati sono persistiti da altri elementi architetturali

DEPLOY AUTOMATIZZATO DI INFRASTRUTTURE A VM

- Definizione di una *GoldenImage* o *Template* come iso base d'installazione
- Deploy automatico e personalizzazioni
  - scripting
  - juju
  - Vagrant
  - Ansible
  - Puppet
  - Chef
  - Mesosphere DC/OS
  - Kubernetes

DEPLOY AUTOMATIZZATO DI INFRASTRUTTURE A CONTAINER

SCRIVO UN *DOCKER-COMPOSE.YML*

Docker, *out-of-the-box* si occupa di implementarlo.

DA *VM* A *CONTAINER*

RICHIEDE UN PO' DI ESERCIZIO ED UN DIVERSO APPROCCIO MENTALE.

- No Gestione Manuale!!! -> Lavoro che andrà perso!
- No Gestione Network!!! -> L'infrastruttura è autonoma in questo!
- No Configurazioni Arlecchino!!! -> Creano *Single-Point-Of-Failure*!
- No Know-how Silos!!! -> Quando serve il *Master-of-Know-how* non si trova!


QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation

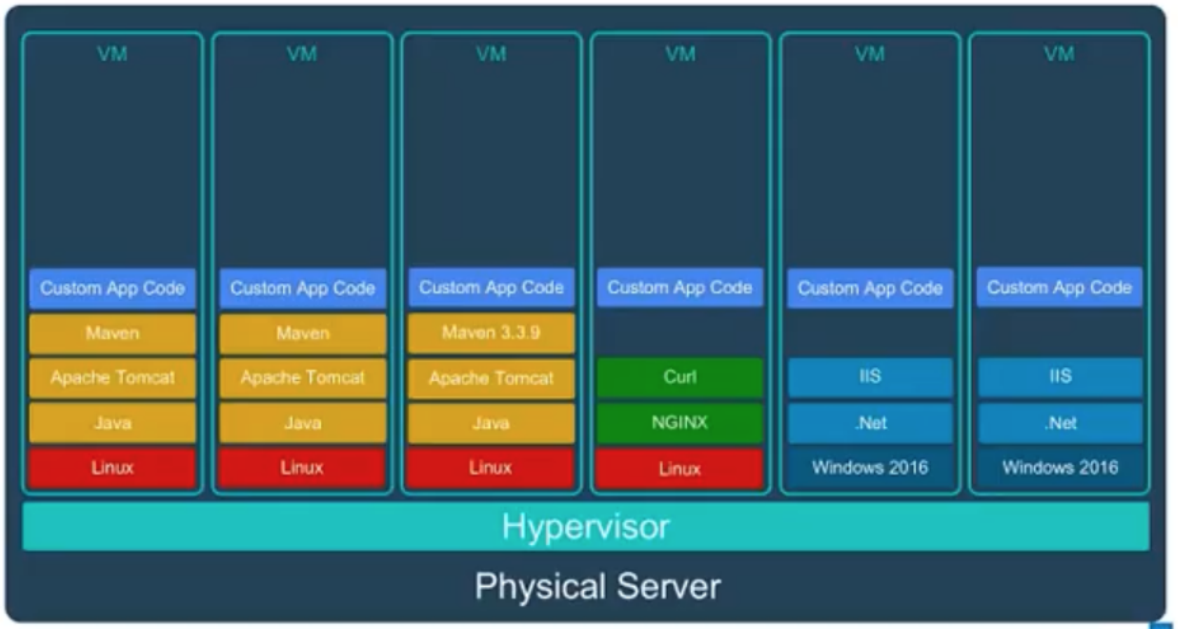
DALLA VM AI SERVIZI

UN MODO PER RIMESCOLARE LE CARTE A NOSTRO VANTAGGIO

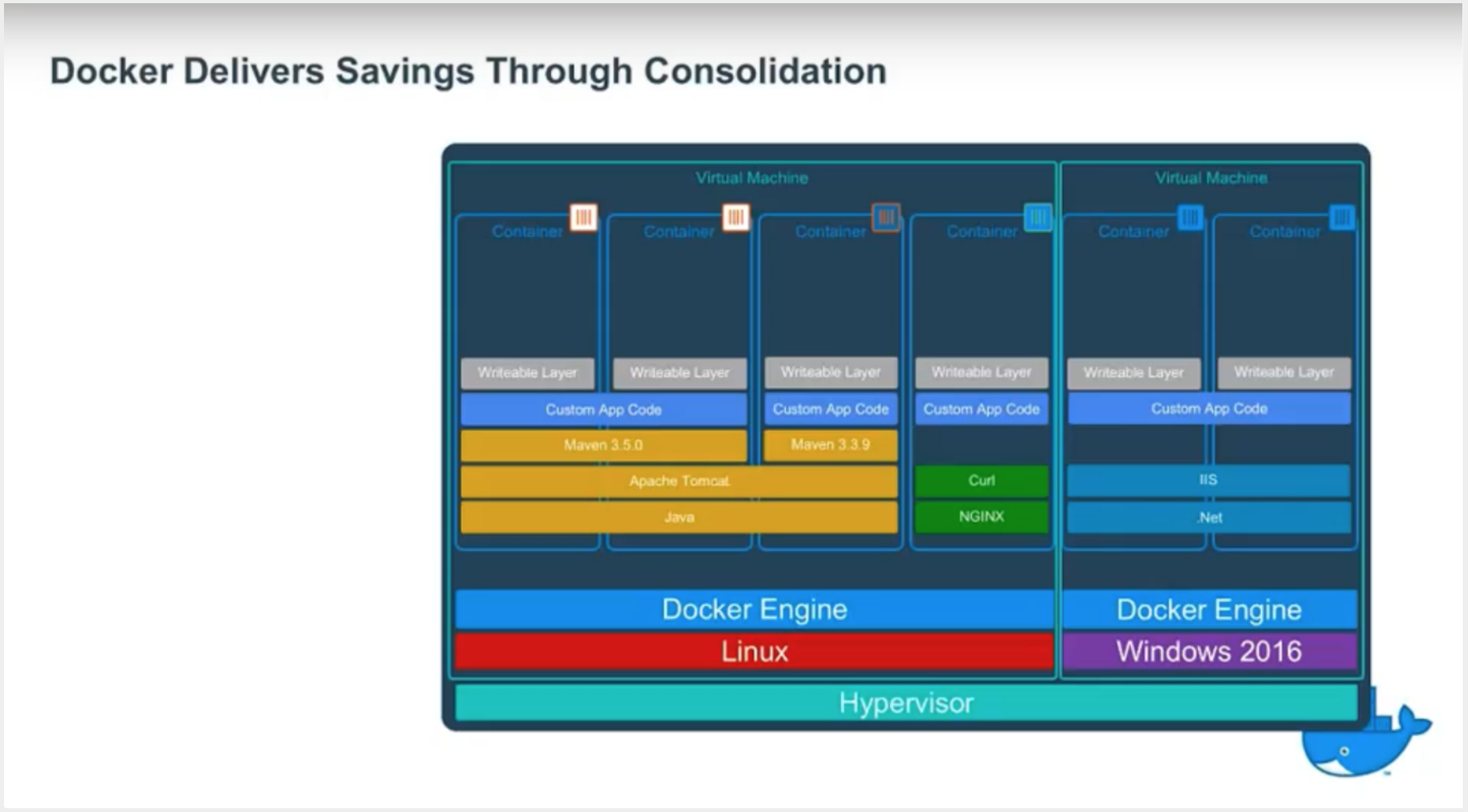
ARCHITETTURA A VM

Our Current State: Virtual Machines

- One app per VM
- Separate OS copy
- Separate app framework
- ”Black box” to hypervisor
- App is tracked elsewhere



ARCHITETTURA A CONTAINER



ANALISI DI UN *DOCKER-COMPOSE.YML*

```yaml
version: '3'
services:
  db:
    image: mysql
    restart: always
    hostname: db
    domainname: develop.local
    container_name: db
    environment:
      MYSQL_ROOT_PASSWORD: secretadmin
    ports:
    - "3316:3306"
    volumes:
    - dbvolume01:/var/lib/mysql
  mail:
```

From: ./04_VM2Servizi/code/01/dockerfile


QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation

SVILUPPO SOFTWARE

ESEMPIO D'USO DI DOCKER NELLO SVILUPPO DI UNA PROCEDURA PHP

OBIETTIVO: CONTAINER COME VIRTUALENV PER PHP

Non installare la versione dell'interprete php necessario allo sviluppo, ma usare l'immagine PHP ufficiale per avere un Container interattivo nel proprio sistema.

PASSI DA SEGUIRE

- Analizzare la configurabilità dell'immagine ufficiale di *PHP* su *DockerHub*
- Analizzare come la propria IDE s'interfaccia all'interprete *PHP* e a *Docker*
- Con una serie di test e prove arrivare alla propria *Comfort-Zone*

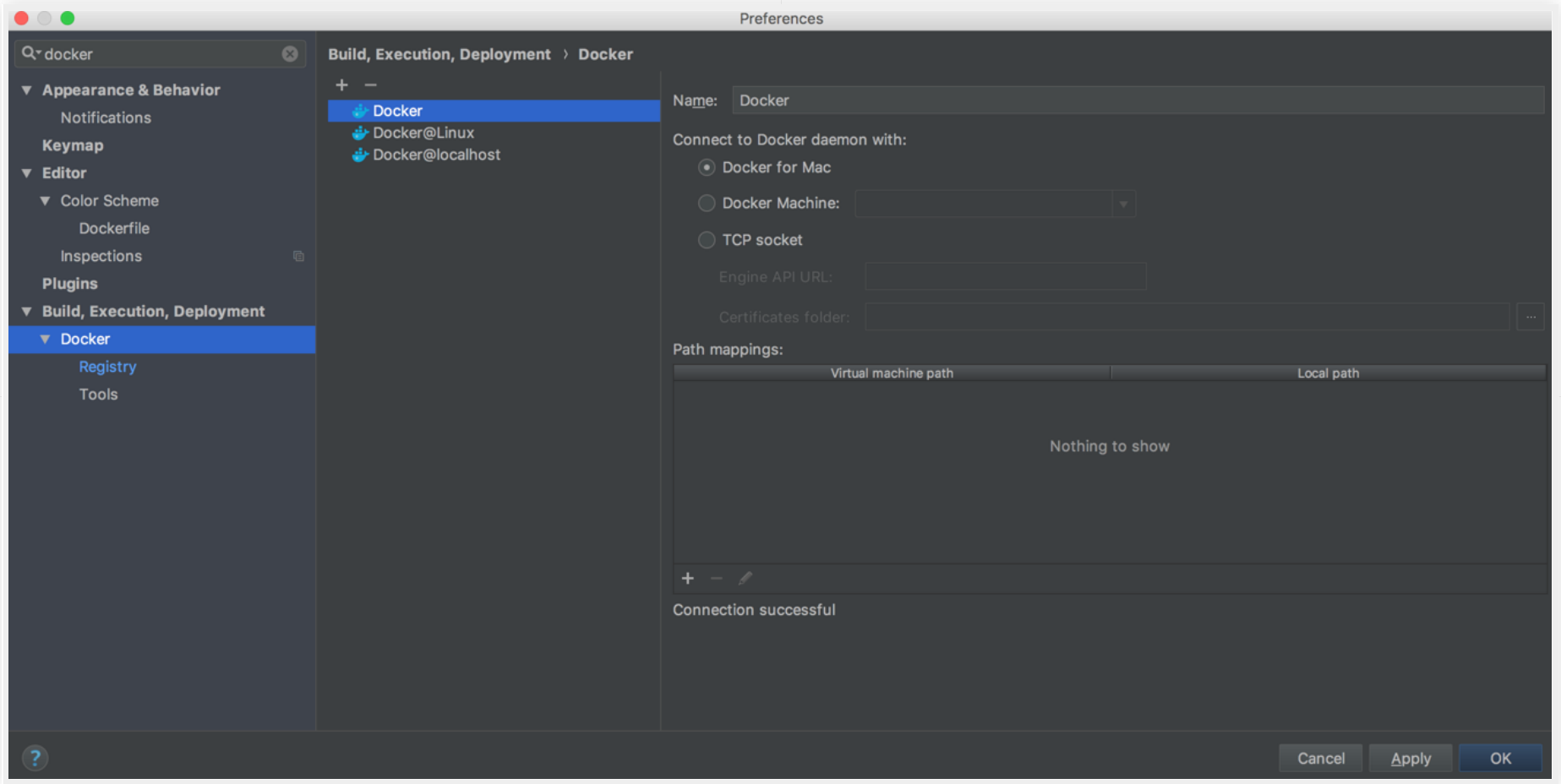
CONFIGURAZIONE SU VSCODE 1/2

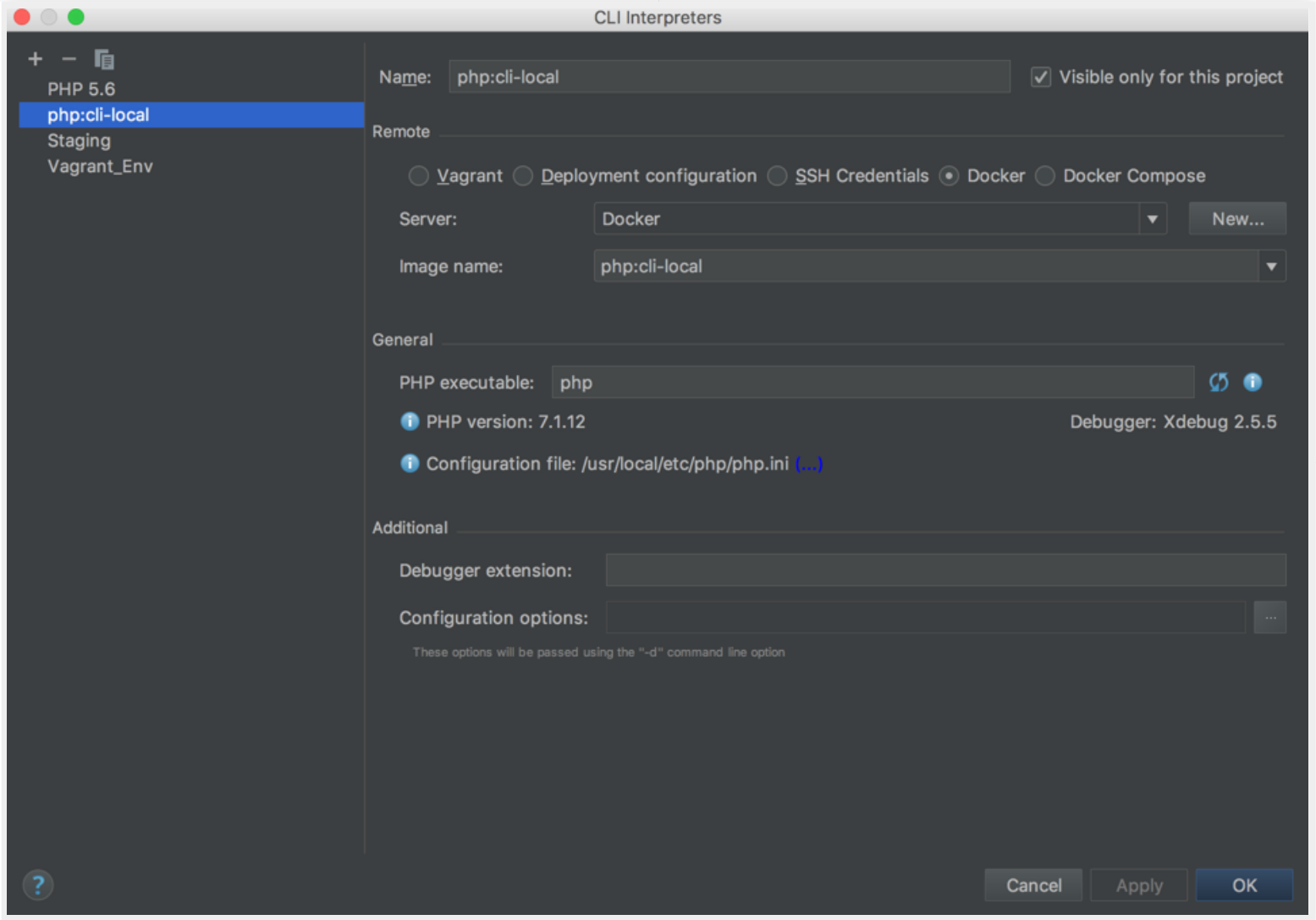
```
{
    "php.validate.enable": true,
    "php.validate.executablePath": "${rootPath}/.vscode/bin/php.sh",
    "php.validate.run": "onSave",
    "php.suggest.basic": false
}       Giuliano Latini, 3 months ago • Create Workspace Settings for PHP
```

CONFIGURAZIONE SU VSCODE 2/2

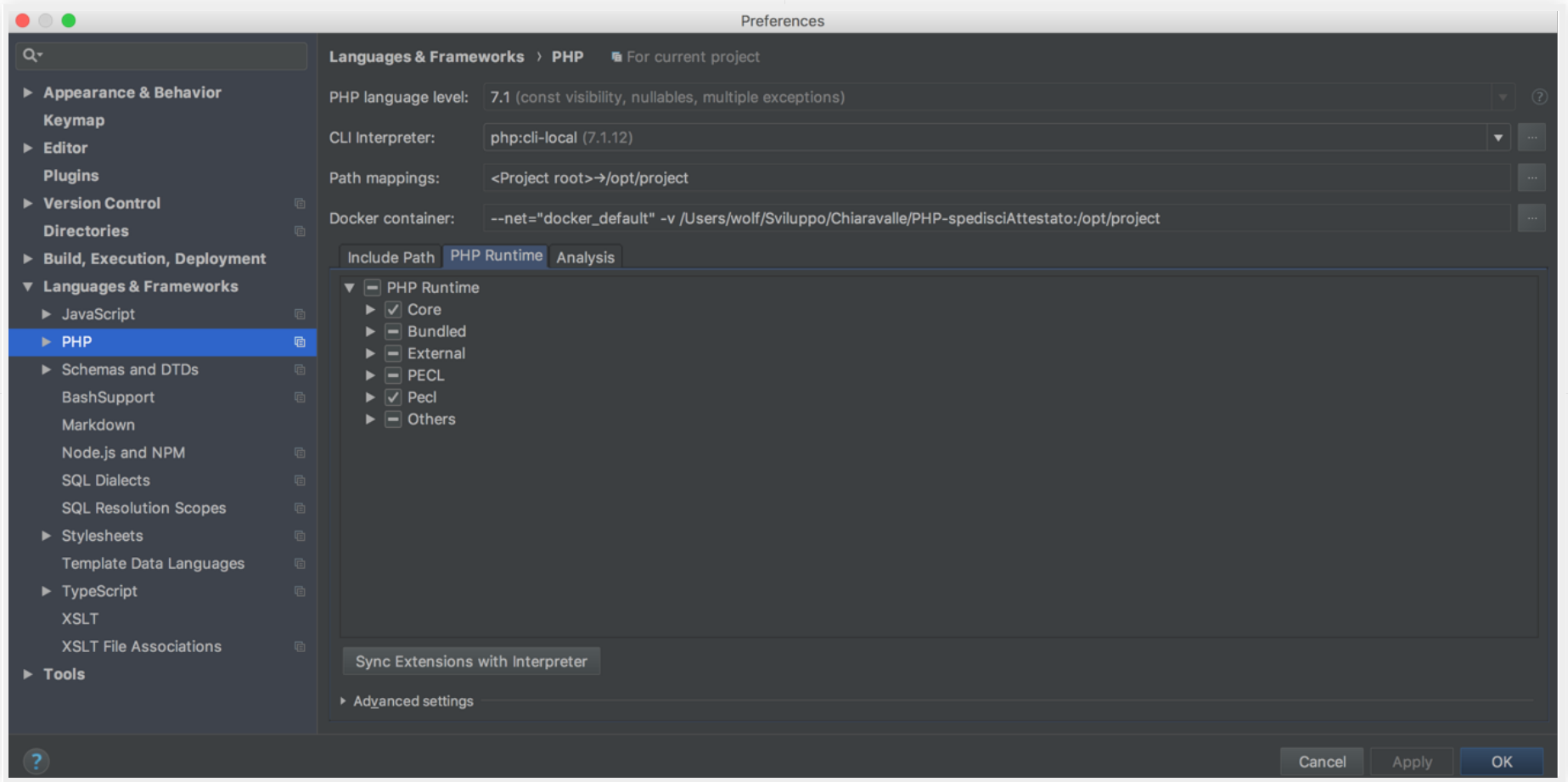
```
#!/bin/sh
# This is a comment!
_heads=${@:1:$(($# - 1))}
_tail=${@:$#}
_tail=${_tail#$(pwd)}
_tail=${_tail#\/}
/usr/local/bin/docker run -i --name php_cli \
--rm -v $(pwd):/app -w /app php:cli-local php $_heads $_tail
```

CONFIGURAZIONE SU PHPSTORM 1/3




CLI Interpreters
PHP 5.6
php:cli-local
Staging
Vagrant_Env
Name:
php:cli-local
Visible only for this project
Remote
Vagrant
Deployment configuration
SSH Credentials
Docker
Docker Compose
Server:
Docker
New...
Image name:
php:cli-local
General
PHP executable:
php
PHP version: 7.1.12
Debugger: Xdebug 2.5.5
Configuration file: /usr/local/etc/php/php.ini (...)
Additional
Debugger extension:
Configuration options:
These options will be passed using the "-d" command line option
Cancel
Apply
OK

CONFIGURAZIONE SU PHPSTORM 3/3




QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation

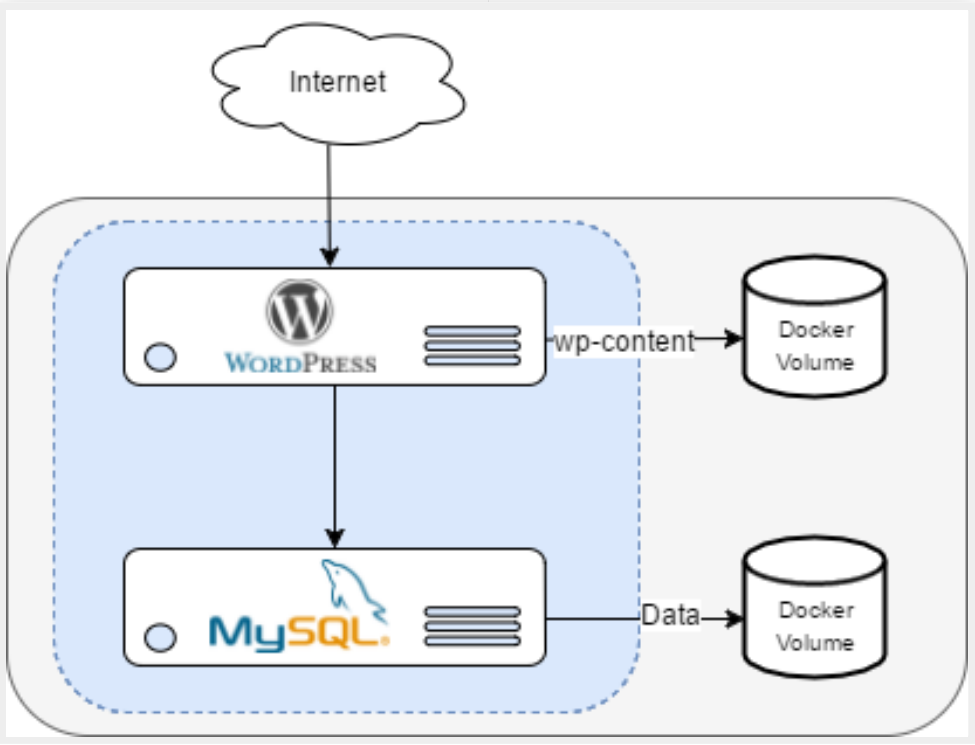
GESTIONE DI SISTEMI

ESEMPIO DI UN'INFRASTRUTTURA WORDPRESS ORGANIZZATA A SERVIZI

DEFINIZIONE FORMALE IMPLEMENTATA CON DOCKER

WORDPRESS IN DOCKER

SCHEMA GENERALE

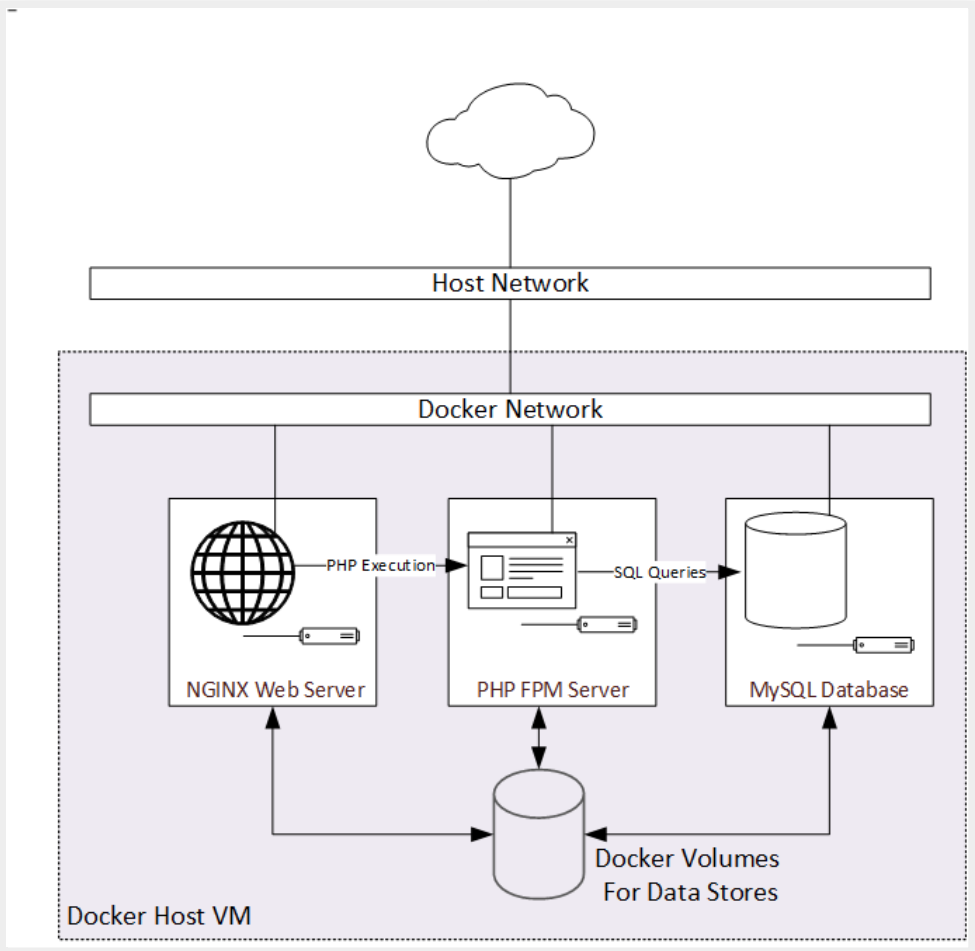


SCHEMA GENERALE (DOCKER-COMPOSE.YML)

```
version: '2'
services:
  wordpress:
      image: wordpress
      links:
          - mysql
      environment:
          - WORDPRESS_DB_PASSWORD=secretp@ssword
          - VIRTUAL_HOST=example.com
      expose:
          - 80
  mysql:
      image: mysql:5.7
      environment:
          - MYSQL_ROOT_PASSWORD=secretp@ssword
```

From: ./06_GestioneSistemi/code/01/docker-compose.yml

SCHEMA IMPLEMENTATIVO

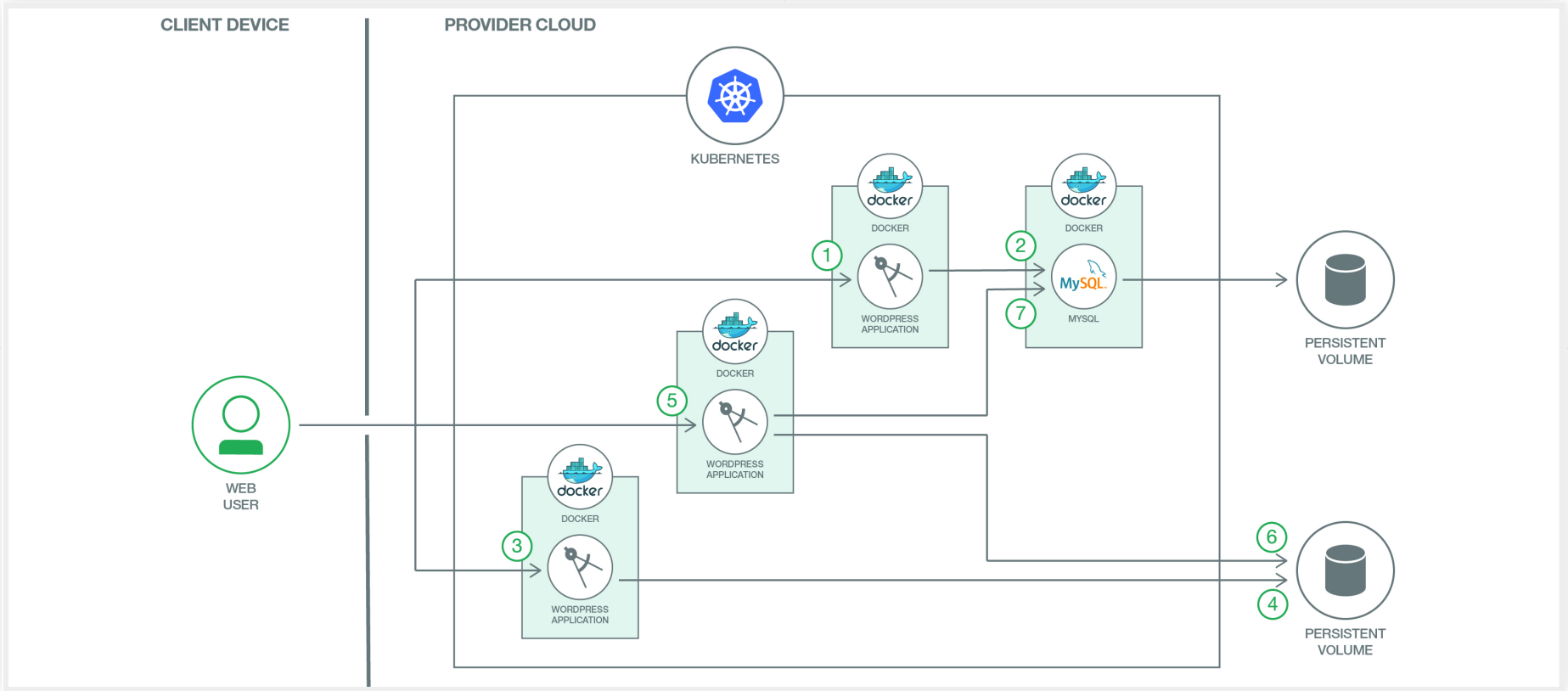


SCHEMA IMPLEMENTATIVO (DOCKER-COMPOSE.YML)

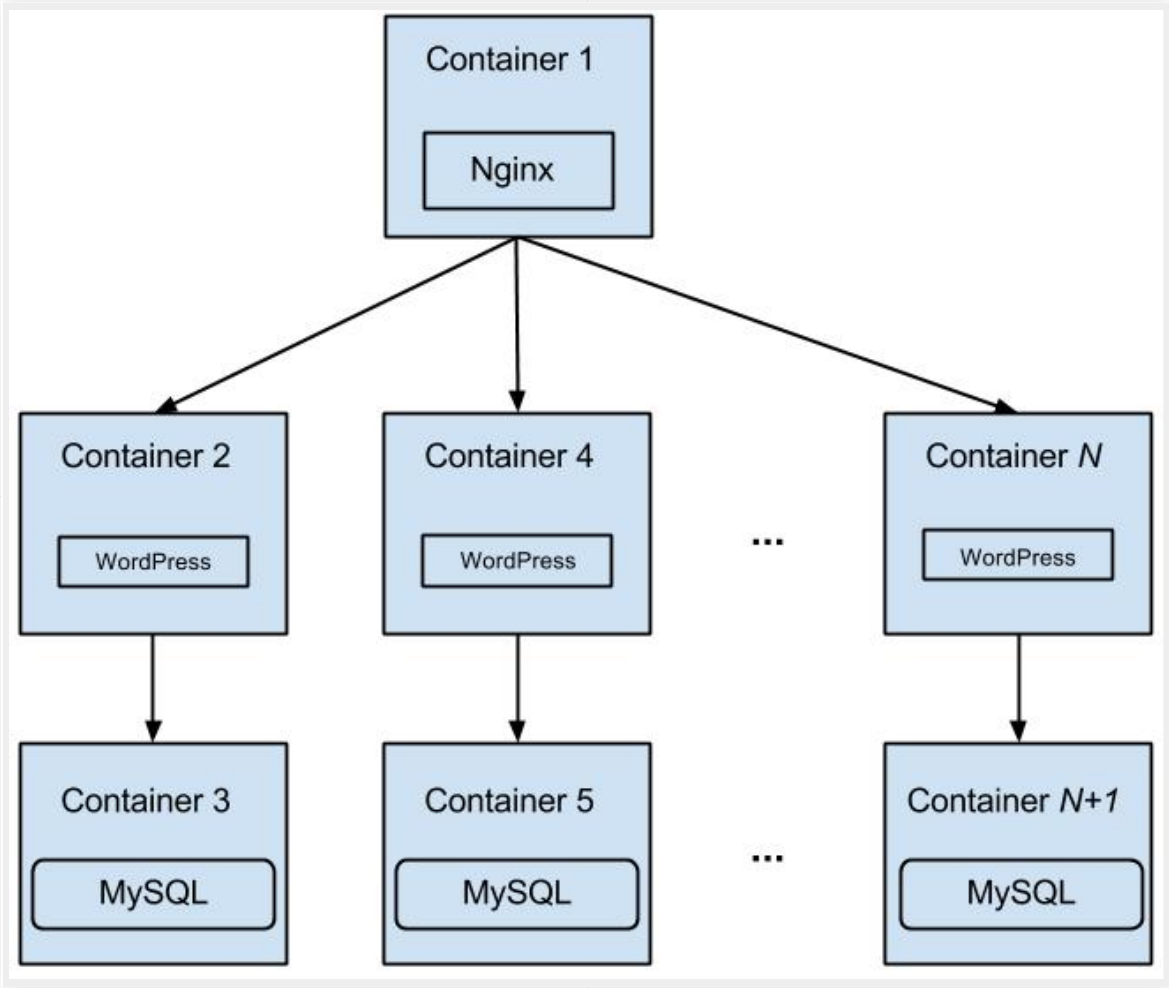
```
version: '3'
services:
    web:
        image: nginx:alpine
        volumes:
            - "./etc/nginx/default.conf:/etc/nginx/conf.d/default
            - "./etc/ssl:/etc/ssl"
            - "./web:/var/www/html"
            - "./etc/nginx/default.template.conf:/etc/nginx/conf.
        ports:
            - "8000:80"
            - "3000:443"
        environment:
            - NGINX_HOST=${NGINX_HOST}
        command: /bin/sh -c "envsubst '$$NGINX_HOST' < /etc/nginx
```

From: ./06_GestioneSistemi/code/02/docker-compose.yml

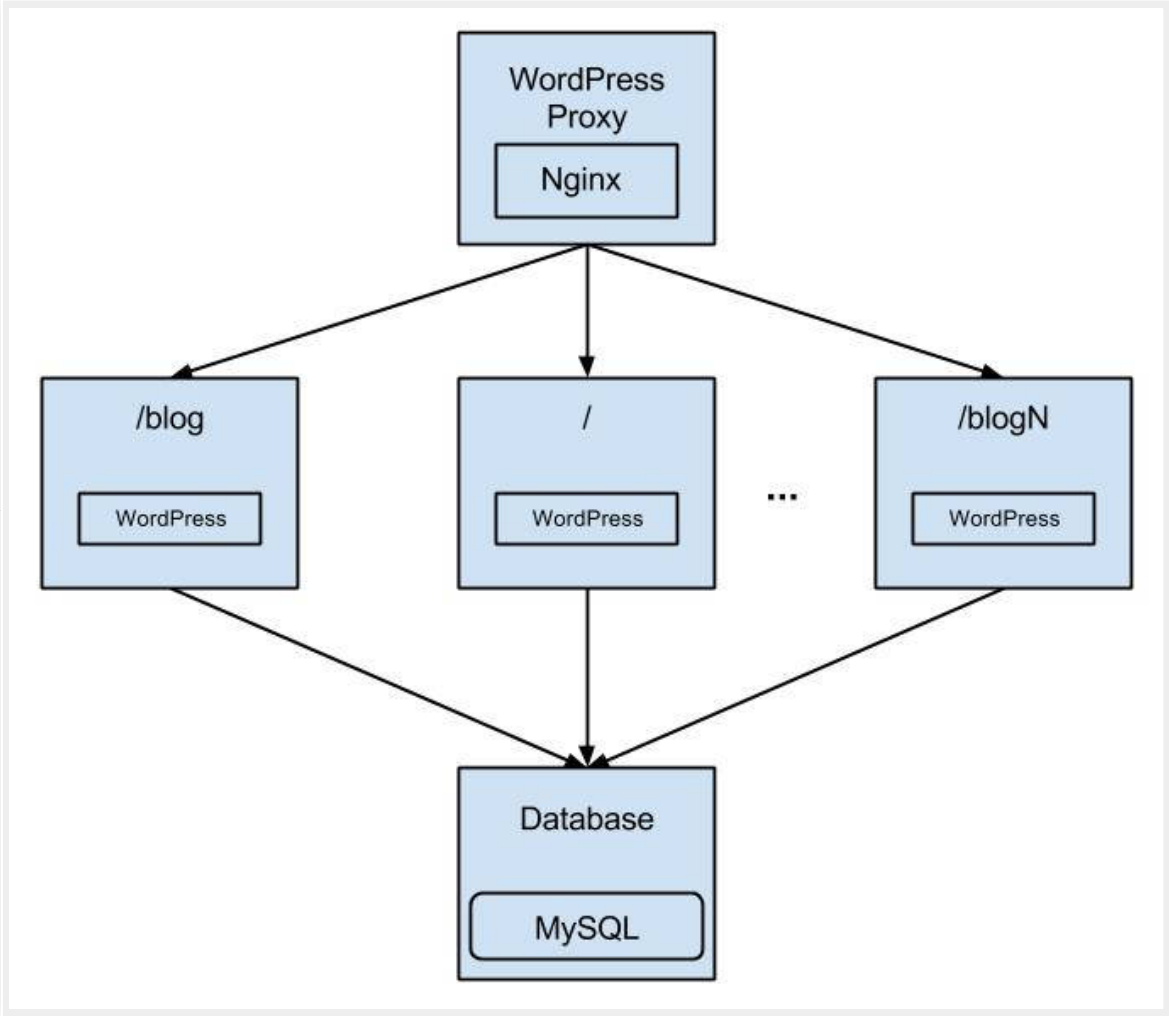
ARCHITETTURA SCALABILE IN DINAMICO



PATTERN ARCHITETTURALE MULTIDBMS



PATTERN ARCHITETTURALE MONODBMS




QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation

DOCKER SWARM

LA CLUSTERIZZAZIONE ED ALTA AFFIDABILITÀ

COME IL SISTEMA RISPONDE E SI ADATTA ALLE SOLLECITAZIONI ESTERNE IN UN SISTEMA DI CALCOLO

SERVIZIO CLOUDGARR

IL CLOUD PER LA COMONITÀ GARR

GARR Federated Cloud

Giuseppe Attardi

giuseppe.attardi@garr.it

Dipartimento CSD

Consortium GARR

Demo

Compute Apps Documentation Community Support Login

GARR Federated Cloud

The GARR Federated Cloud offers cloud services to the Italian academic and research community. GARR coordinates a federation of clouds, located in national datacenters owned by members of the GARR community, which participate to the federation by sharing resources and services.

The GARR cloud allows creating and managing Virtual Machines (IaaS) as well as stacks and applications (PaaS).

Go to the dashboard

Virtual Machines

The GARR Cloud delivers **virtual machines** running in the data centers of the *GARR Federated Cloud* connected through the GARR high speed fiber network. The GARR Cloud provides tools for self provisoning computing resources and deploying applications and services, enabling scaling from single instances to clusters of integrated and load-balanced cloud computing.

Register to GARR Federated Cloud

Sign up with IDEM or eduGAIN

Sign up with Google

Create a new account

Federated Cloud

Objectives

- Facilitate **transition towards cloud computing**
- Allow **resource sharing**, maintaining **control of use**
- Exchange **best practices** on management and use
- Evolve towards **native cloud applications**
- Expand **catalogue of cloud applications**

GARR Commitments

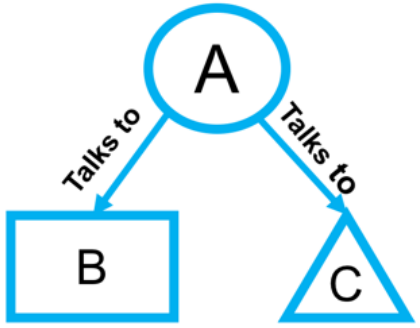
- Architecture Design
- Ready to use OpenStack Distro
- OpenSource Code Base (git.garr.it)
- Upgrades and Maintenance
- Solution for multiple tenancy
- Federation and Delegation
- Federated Authentication
- Asset Management

TCO: after ownership

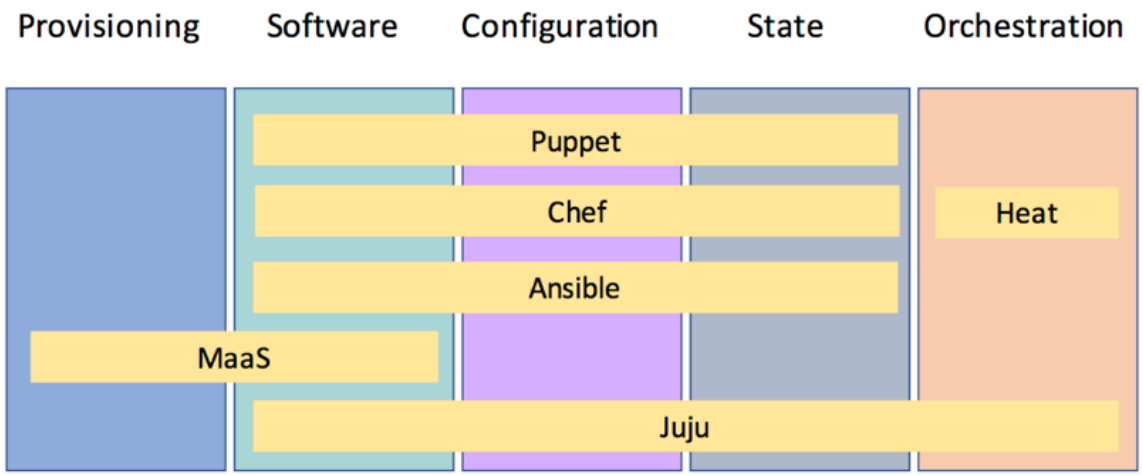
- If you buy services, at the end **what does it remain to you**?
  - Just the experience on using somebody else's product
- If you build your own infrastructure, at the end what remains to you?
  - The infrastructure
  - The experience in building the infrastructure
  - The experience on using the infrastructure

Declarative Modeling

- App A requires:
  - X GB memory and Y CPU
  - N GB storage
  - talking with B and C
  - An URL endpoint
  - To run locally, close to B

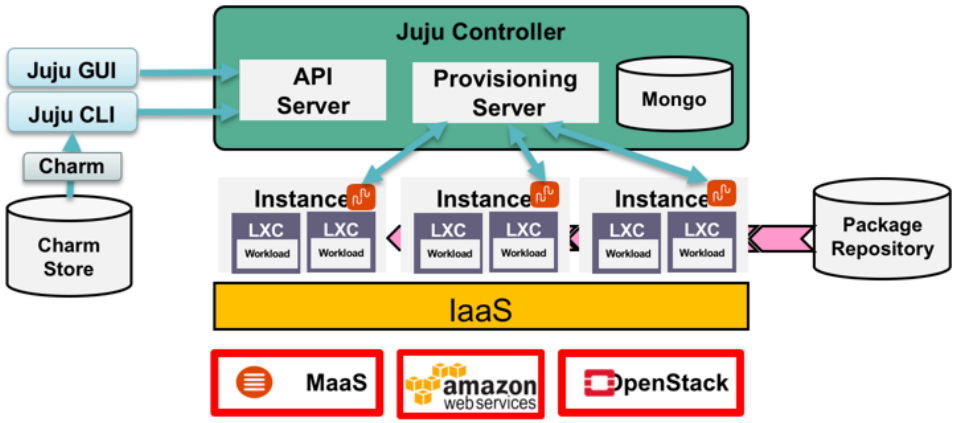


Automation Tools



Juju Architecture and Workflow

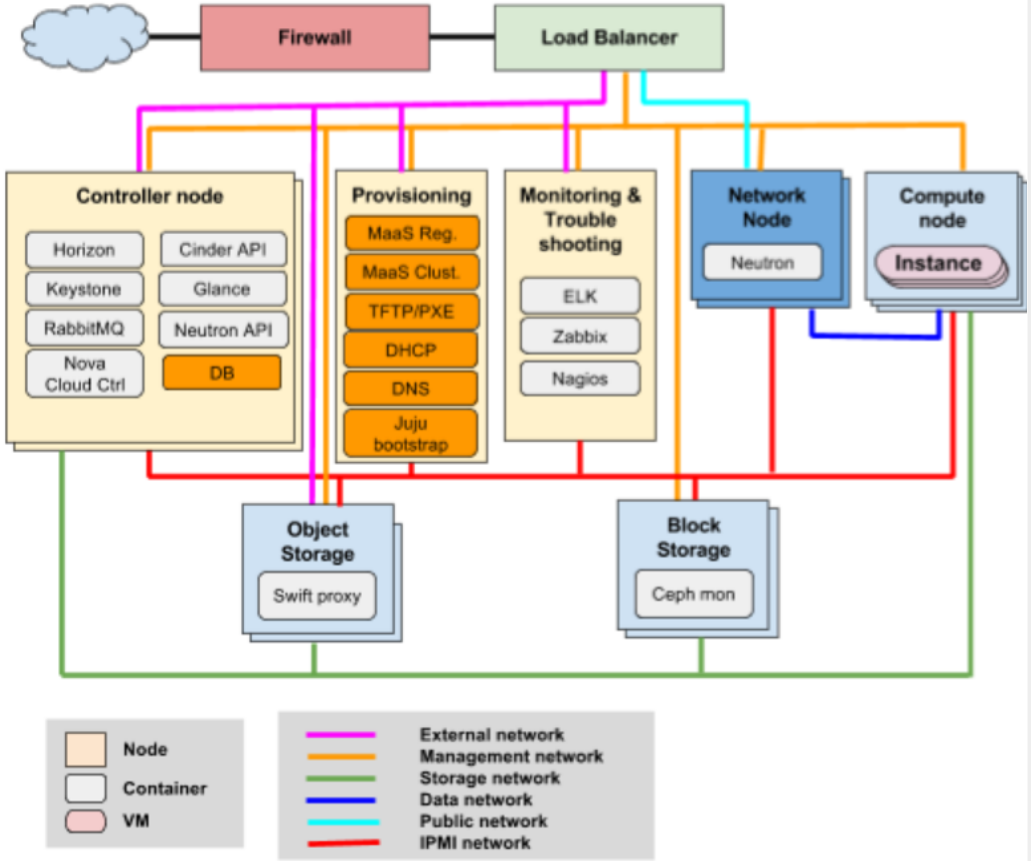
- **Reactive engine**, triggered by events that fire corresponding handlers
- **Multiple handlers** may match and will be run in a undetermined order
- Handlers may cause additional events
- The engine **runs until convergence** to a stable state



OpenStack

GARR Reference Architecture

- Widely used and well supported Cloud Computing software
- Over 45.000 developers world wide
- Complex to manage
- Designed a Reference Architecture:
  - Declarative modeling
  - Easy to configure and to replicate
  - Managed with automated orchestration tools



Status

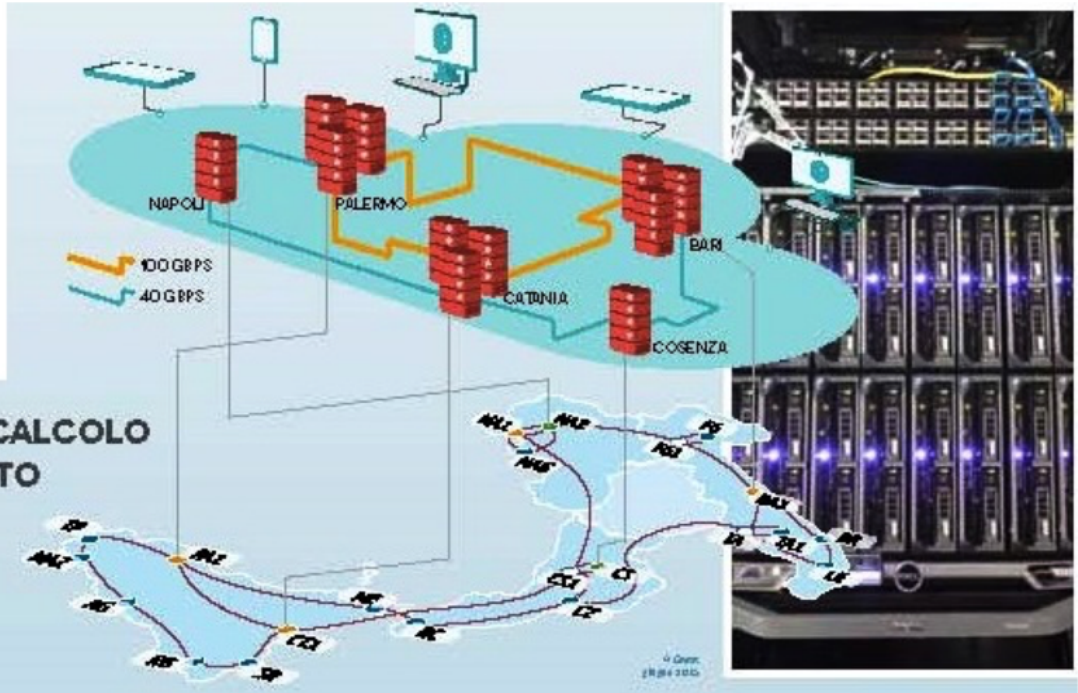
- **Resources**
  - ~9000 vCPU
  - 10 PB Storage
- **Usage**
  - Over 700 users
  - Over 1000 VM
- **Guarantees**
  - Service Continuity
  - Data Protection

Usage

Download CSV Summary (?start=2016-11-01&end=2017-03-22&format=csv

Project Name	VCPUs	Disk	RAM	VCPU Hours	Disk GB Hours	Memory MB Hours
isti	368	2.8TB	656GB	164333.47	1322184.30	279763673.66
unipa	336	7.5TB	1.2TB	693206.14	16207815.52	2692199747.8
ws2017ipv6	128	2.5TB	256GB	14955.82	299116.37	30629516.41
lns-prj1	54	543GB	71.5GB	118279.27	1629777.45	189620873.70
garrdemo318	32	20GB	32GB	16914.14	10571.34	17320083.68
INFN-FI	32	40GB	16GB	601.92	771.89	309779.80
wsosadmin	27	540GB	54GB	4510.02	90200.37	9236517.59
GEANT	25	481GB	48.5GB	29681.28	591502.48	60615623.66
infn-vlabs	24	480GB	48GB	8052.53	161050.61	16491582.24
demo	17	340GB	34GB	9877.99	197559.85	20230128.54
SSSUP	16	80GB	16GB	12651.09	63258.04	12954892.40
garrdemo125	8	100GB	128GB	13429.14	167864.31	220023110.77

Deployment Infrastructure



INFRASTRUTTURA DI CALCOLO E STORAGE DISTRIBUITO

- 5 siti distribuiti
- 8.448 virtual CPU
- 10 PB spazio storage

Container Platform for AI

- Automated deployment on bare metal, AWS or other clouds by Juju
- Workloads deployed by Juju
- Distributed storage system using Ceph
- NFS cluster for sharing big data
- Docker containers managed Kubernetes

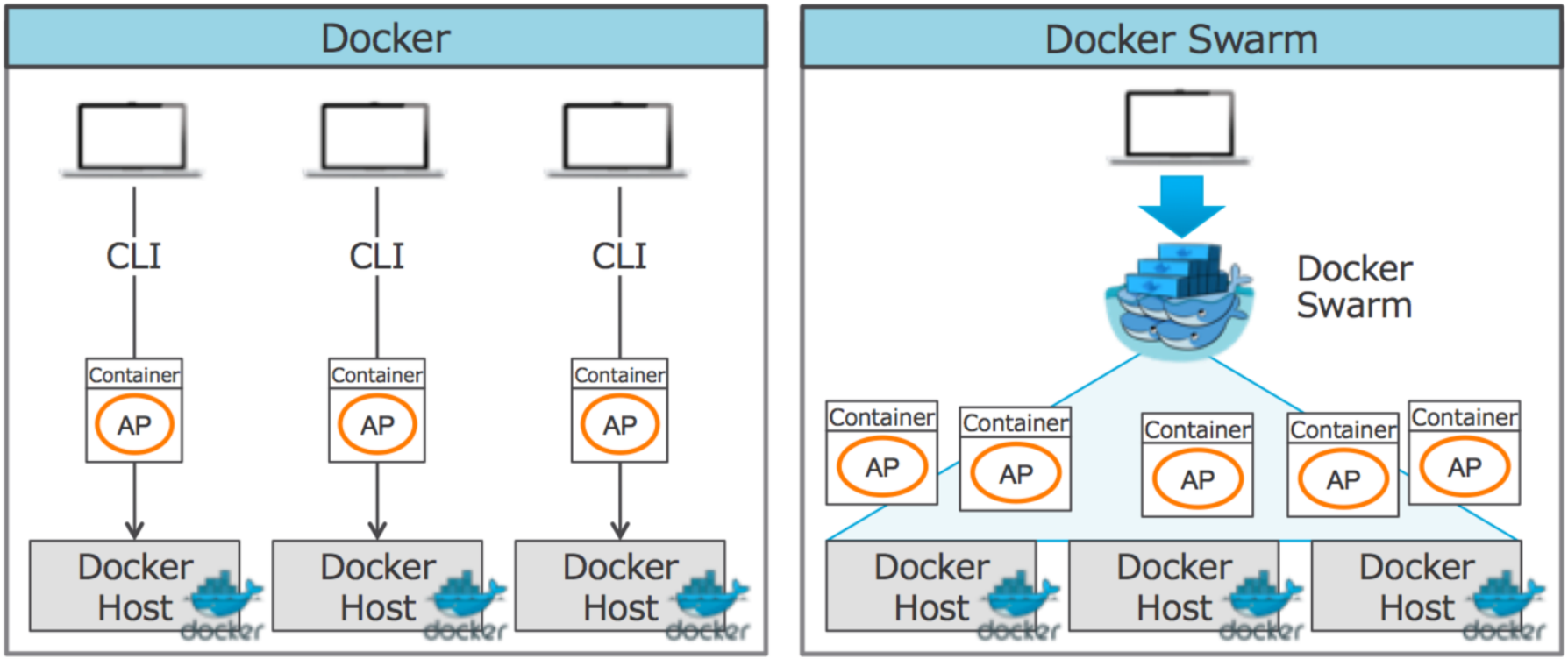
Community

- Build a community of users and developers:
  - https://cloud.garr.it/community/
- Build a shared Catalogue of services
- Examples:
  - Moodle as a Service
  - Jupyter Notebooks as a Service

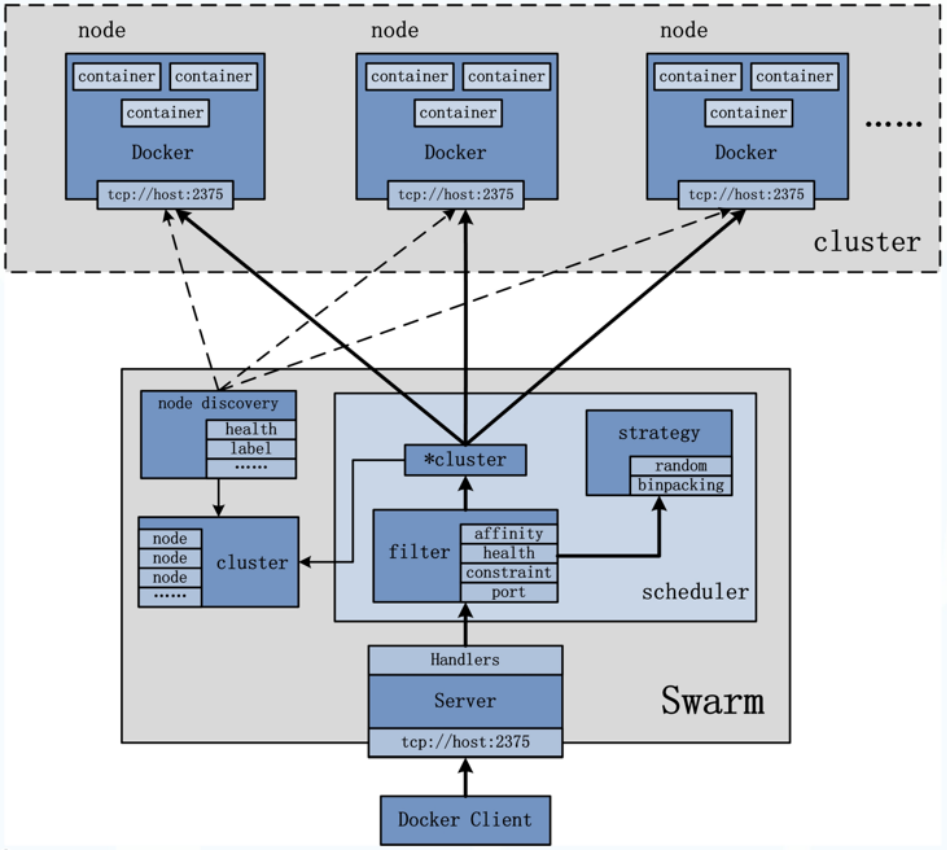
SWARM

CLASTERIZZAZIONE DI RISORSE

Docker Engine vs Docker Swarm

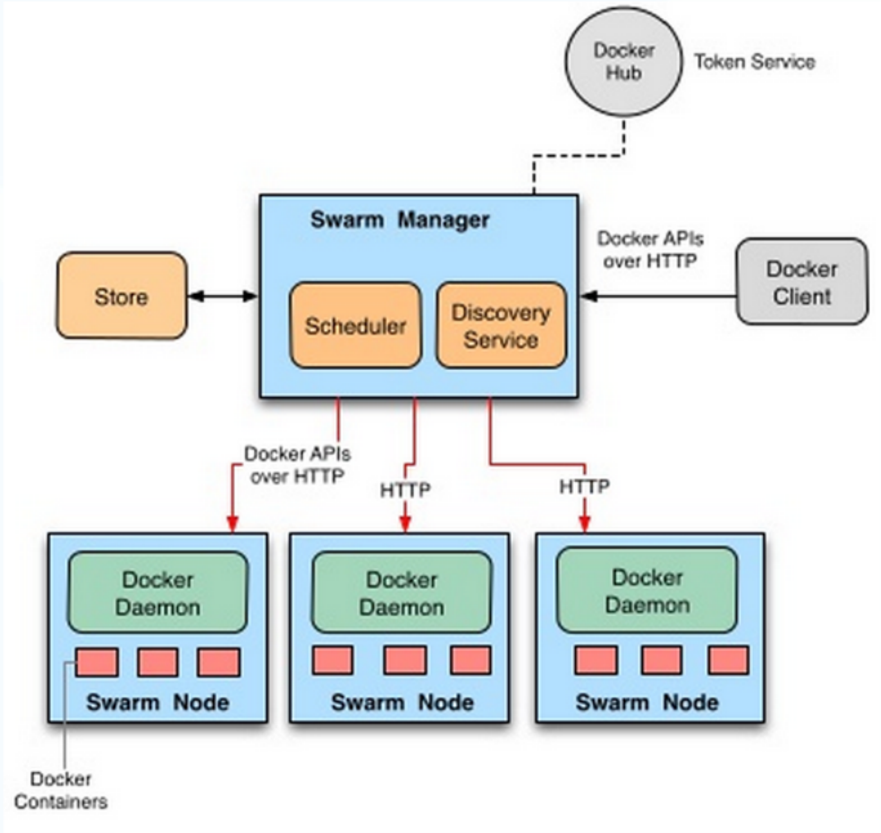


ARCHITETTURA FISICA DOCKER SWARM



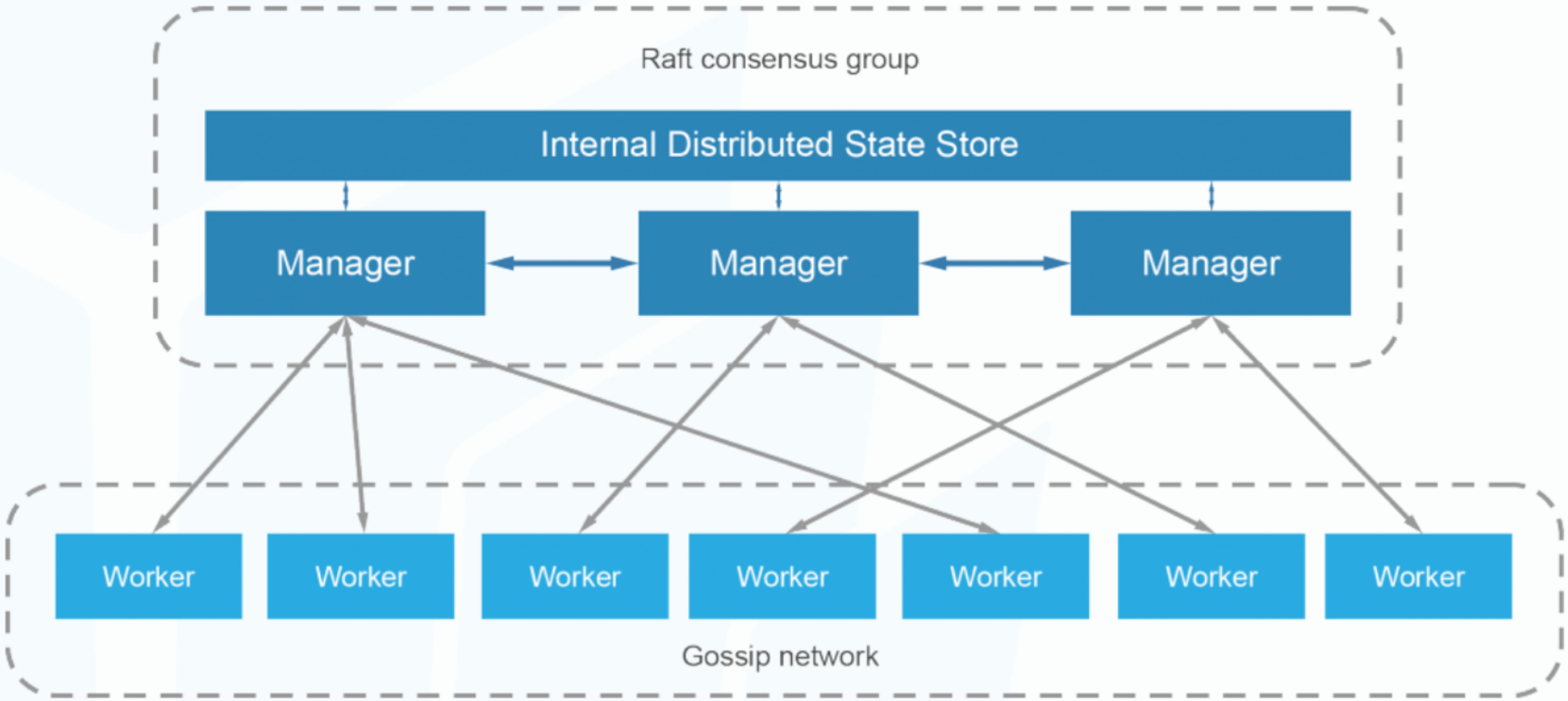
Swarm master organizza una rete di docker-host garantendo raggiungibilità ed accessibilità così da disporre tramite il docker client di un punto di accesso unico a tutte le risorse.

ARCHITETTURA FUNZIONALE DI UN CLUSTER

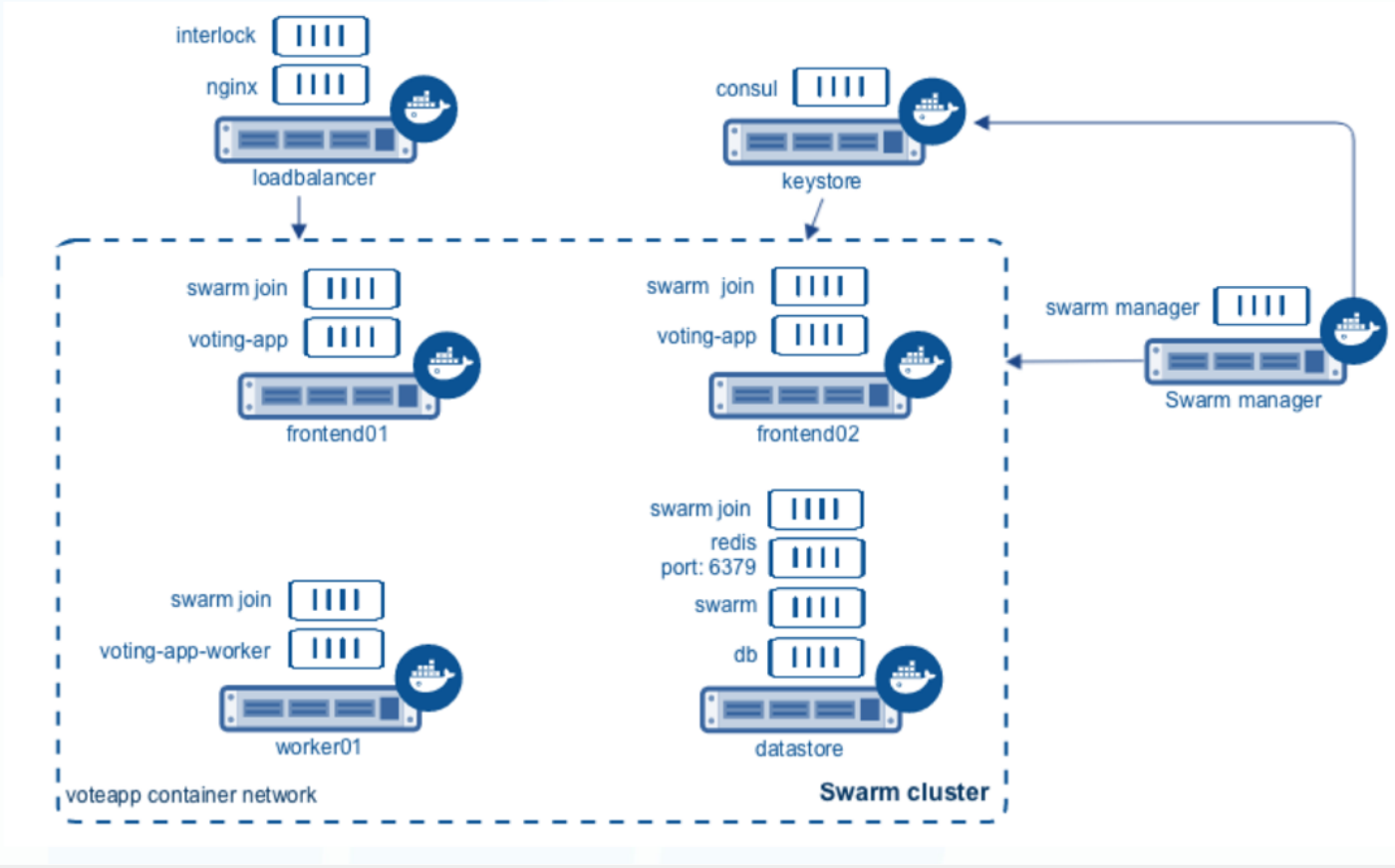


- Docker Client facilità l'accesso alle risorse del cluster.
- Docker Hub è il repository da cui attingere per le immagini da eseguire come containers dal Docker Daemon.

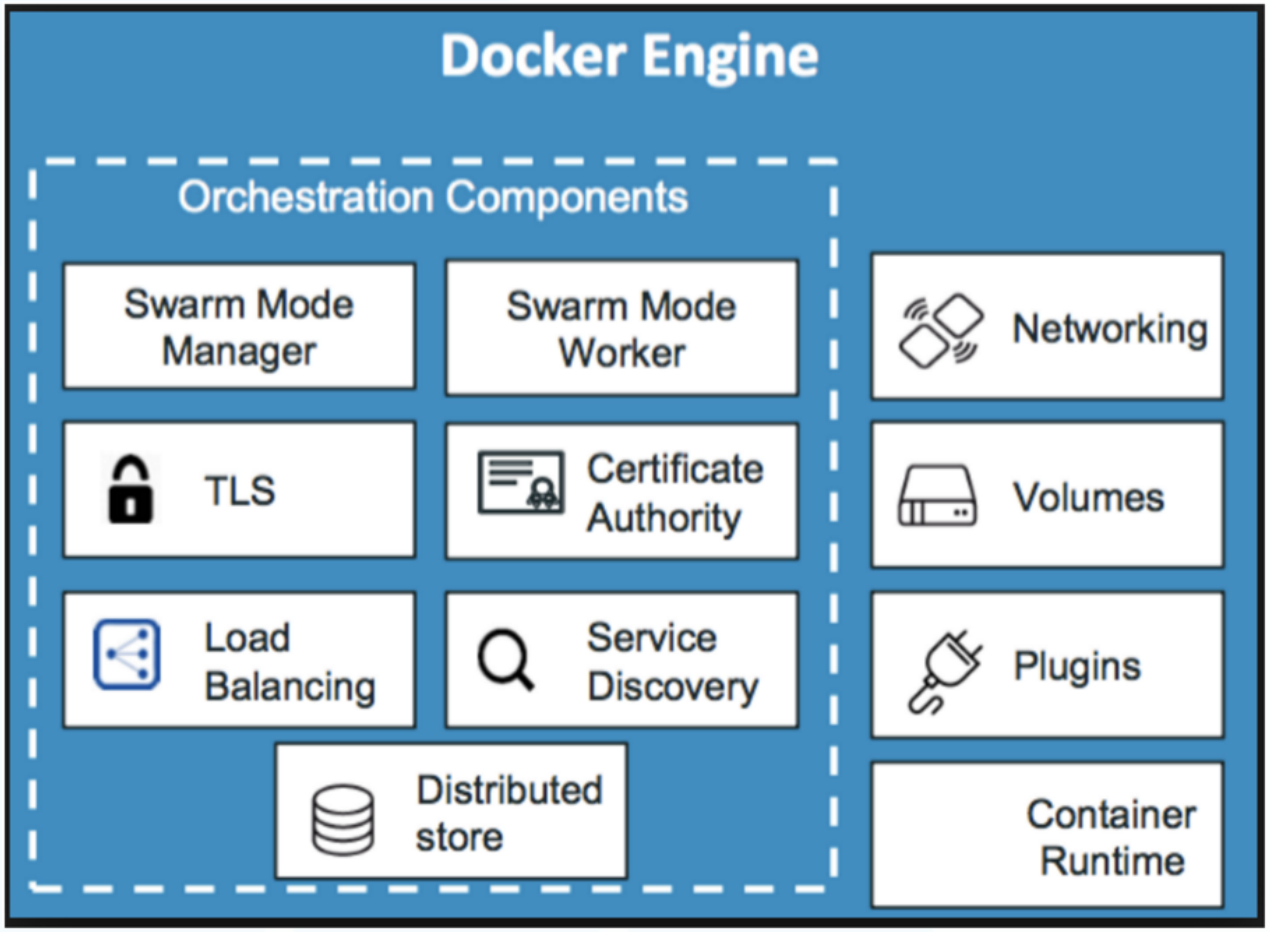
ARCHITETTURA FUNZIONALE DI UN CLUSTER

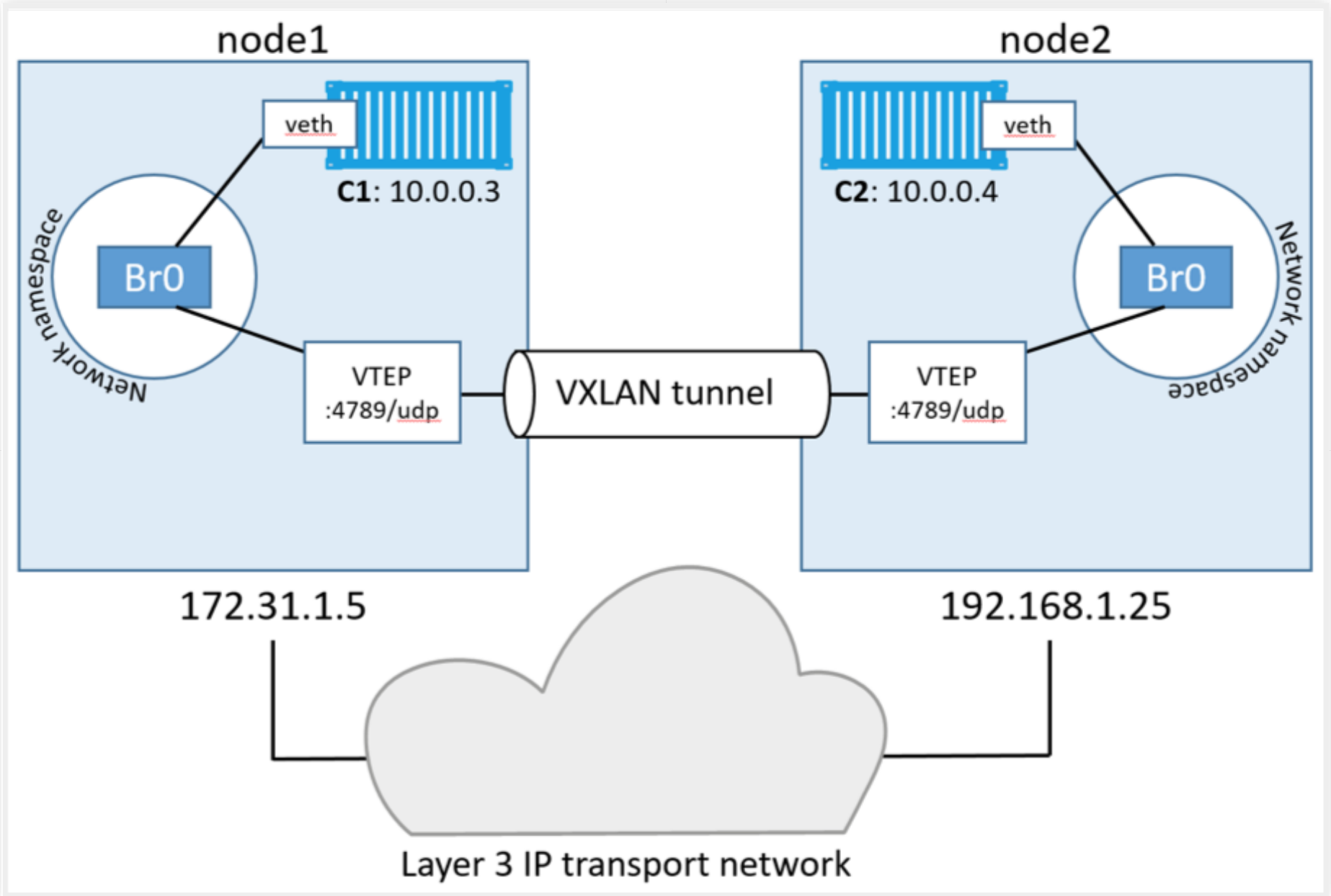


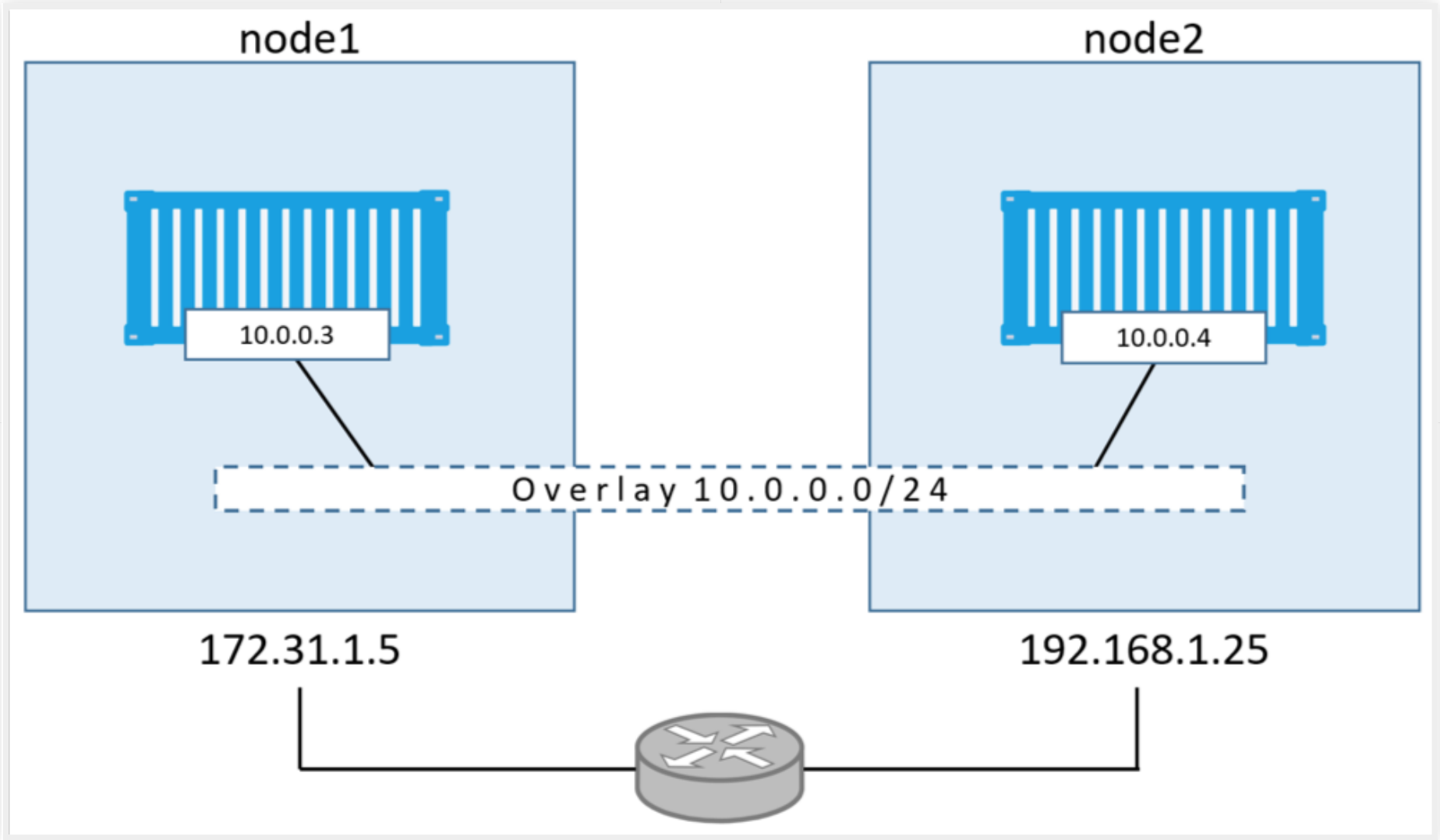
ARCHITETTURA FUNZIONALE DI UN CLUSTER



LA MAGIA IN DOCKER ENGINE




node1
node2
veth
veth
C1: 10.0.0.3
C2: 10.0.0.4
Br0
Br0
Network namespace
Network namespace
VTEP
:4789/udp
VXLAN tunnel
VTEP
:4789/udp
172.31.1.5
192.168.1.25
Layer 3 IP transport network


node1
node2
10.0.0.3
10.0.0.4
Overlay 10.0.0.0/24
172.31.1.5
192.168.1.25


QUESTIONS
how?
discover
questions
where?
?
why
asking questions
challenge
who?
clues
ask
who?
discover
what?
when?
Knowing
investigation
?
clues
how
why?
ask
Knowing
investigation