Network Function Virtualization

Carlo Vallati
Assistant Professor@ University of Pisa
c.vallati@iet.unipi.it

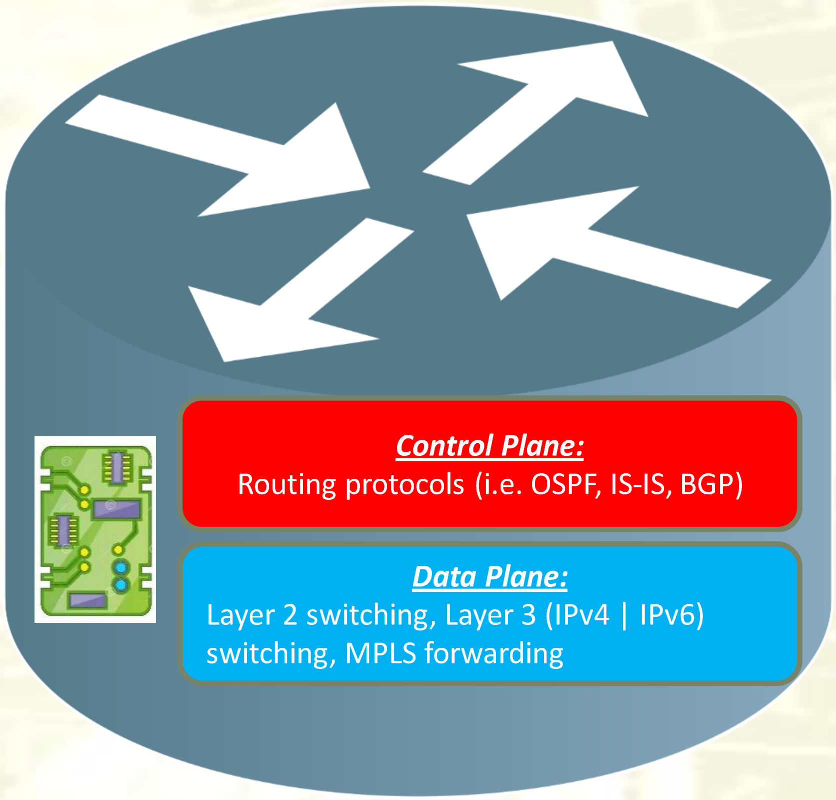
Traditional Network Equipment

- A limited set of functionalities (just a little more than routing/forwarding) implemented in hardware

- Network is designed around the hardware and not viceversa

- Possible changes are limited
- New network services can not be created

- All is fine for the core of ISP networks

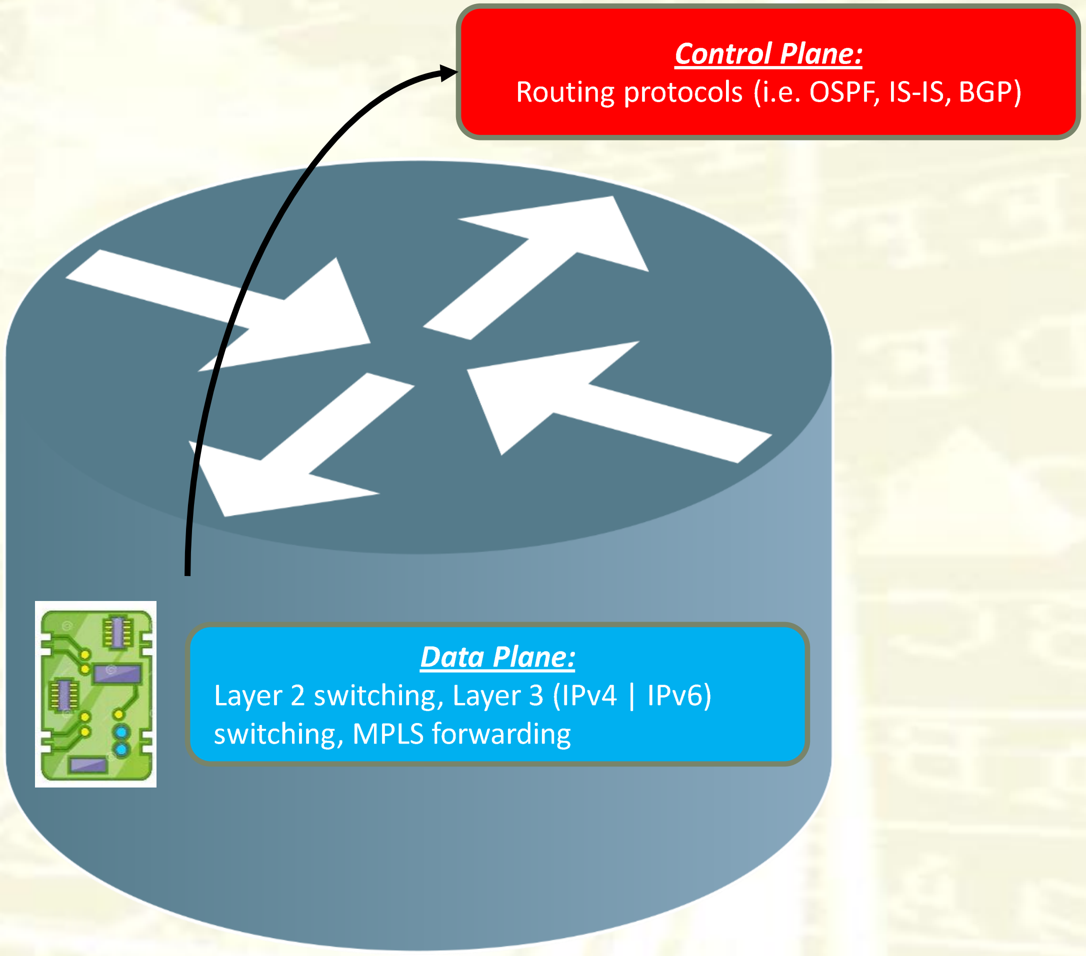

Control Plane:
Routing protocols (i.e. OSPF, IS-IS, BGP)
Data Plane:
Layer 2 switching, Layer 3 (IPv4 | IPv6) switching, MPLS forwarding

SDN Network Equipment

- SDN hardware is highly reconfigurable
- The set of SDN functionalities is small and implemented in hardware

- Again, possible changes are limited
- Again, new network services can not be created if not already there

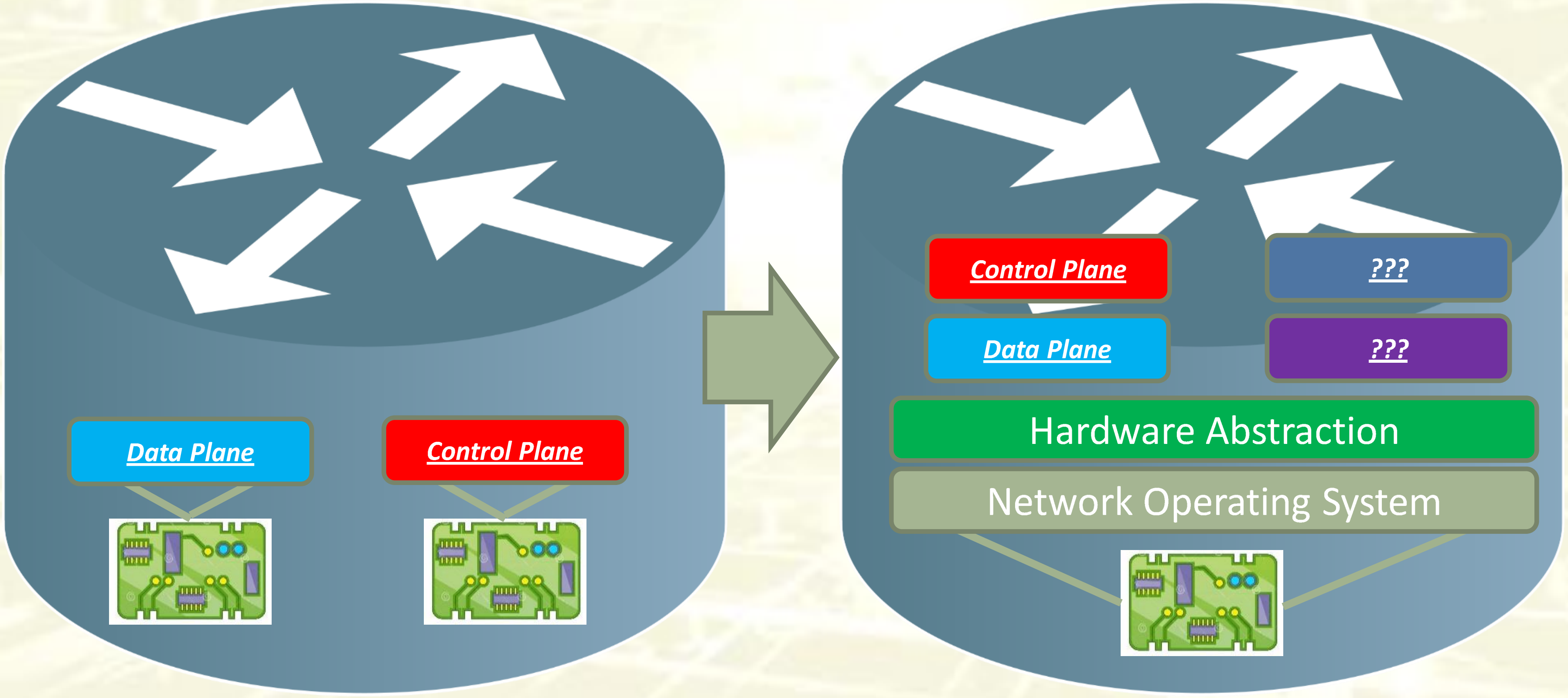
- All is fine at the core of the datacenter where re-configurability is all


Control Plane:
Routing protocols (i.e. OSPF, IS-IS, BGP)
Data Plane:
Layer 2 switching, Layer 3 (IPv4 | IPv6) switching, MPLS forwarding

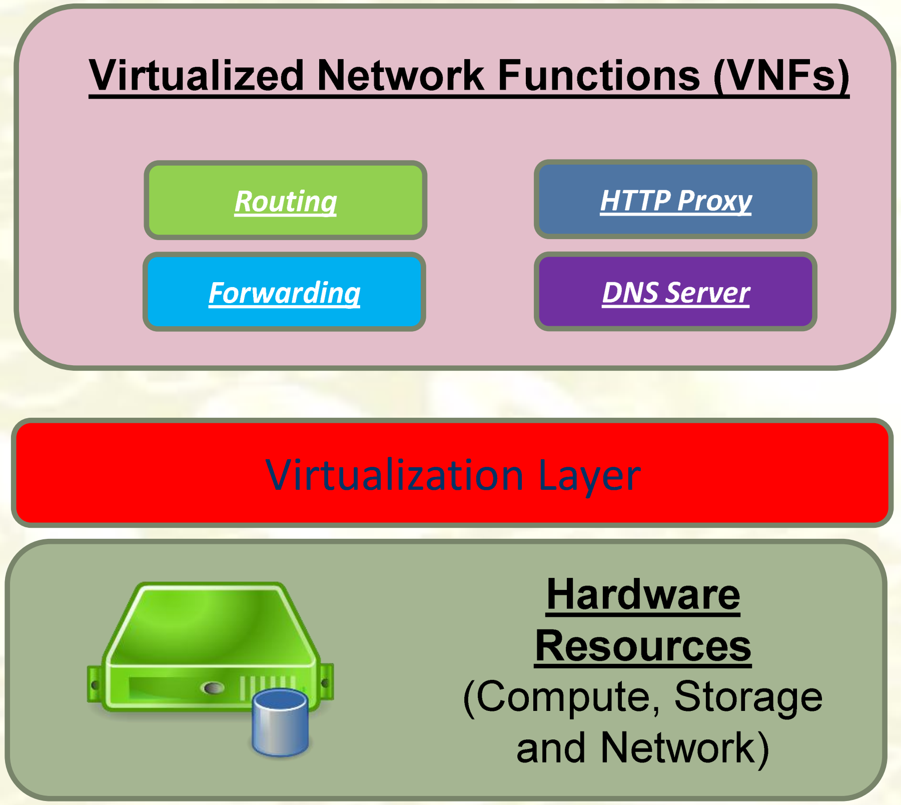
Network Function Virtualization

Motivation: Lack of flexibility / Creation of new services difficult

Solution: Apply the same virtualization paradigm exploited for computing to networking: virtualize the functions of the network; <u>**implement the functions as software-only entities that are designed to be independent from the hardware**</u>



Next generation Network Equipment



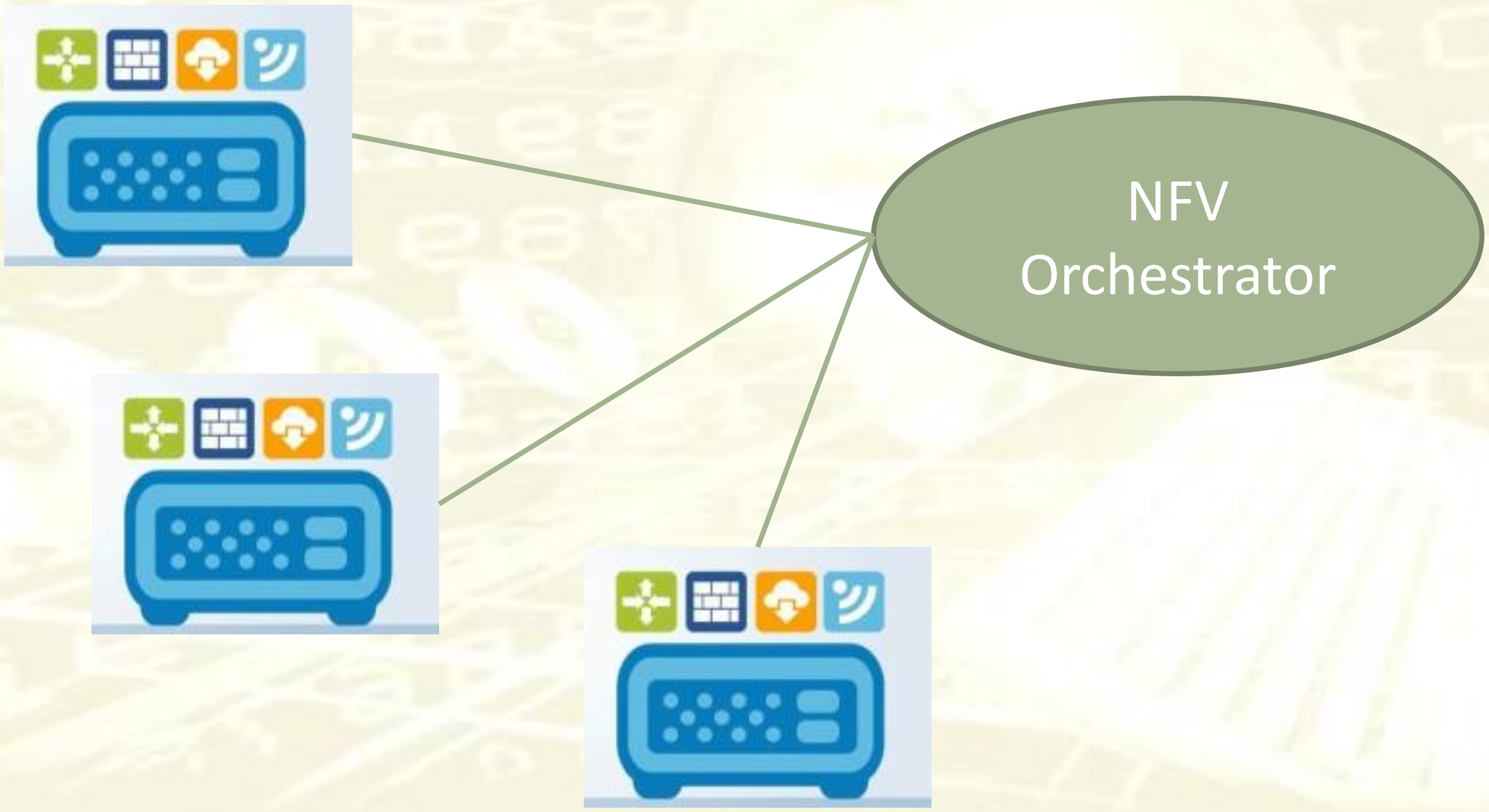
- Run the network software entities on top of off-the-shelf compute and storage elements, using the same **virtualization and cloud technologies** of recent IT infrastructures

- VNF are implemented as a **Virtual Machine** or a **Container**

- Network functions can be deployed at runtime as it is needed

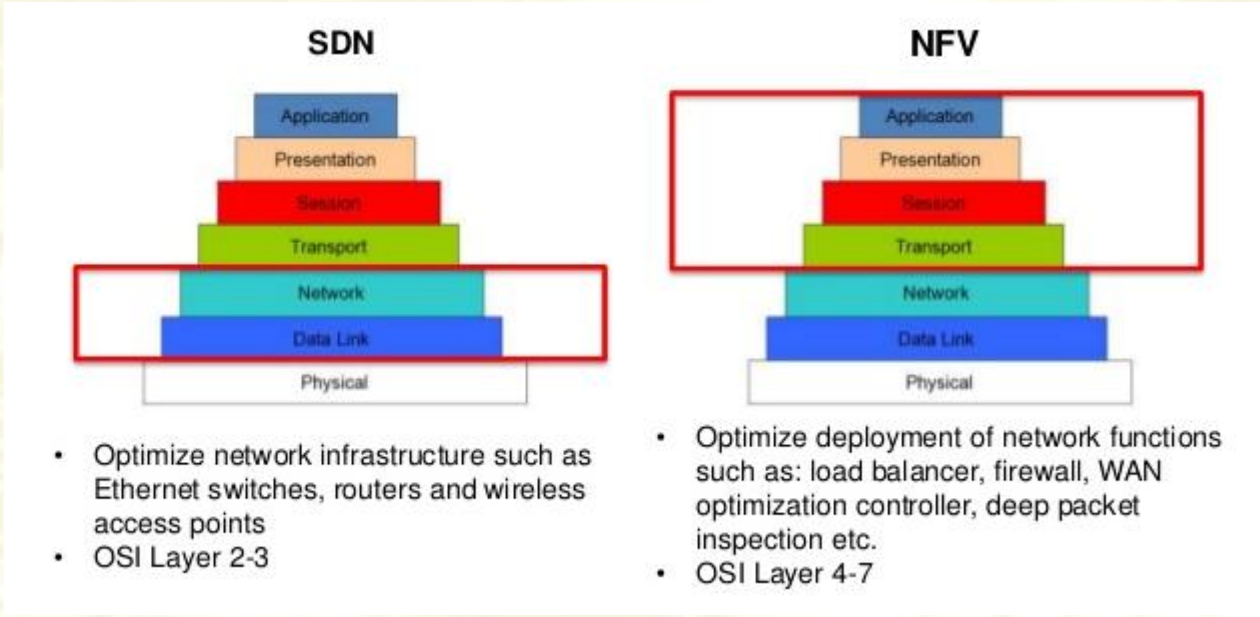
NFV Orchestrator

Softwarization allows to control all the components of each virtualized network equipment by a centralized entity, called **NFV Orchestrator**, *which controls the services to be instantiated on each device*



SDN vs NFV

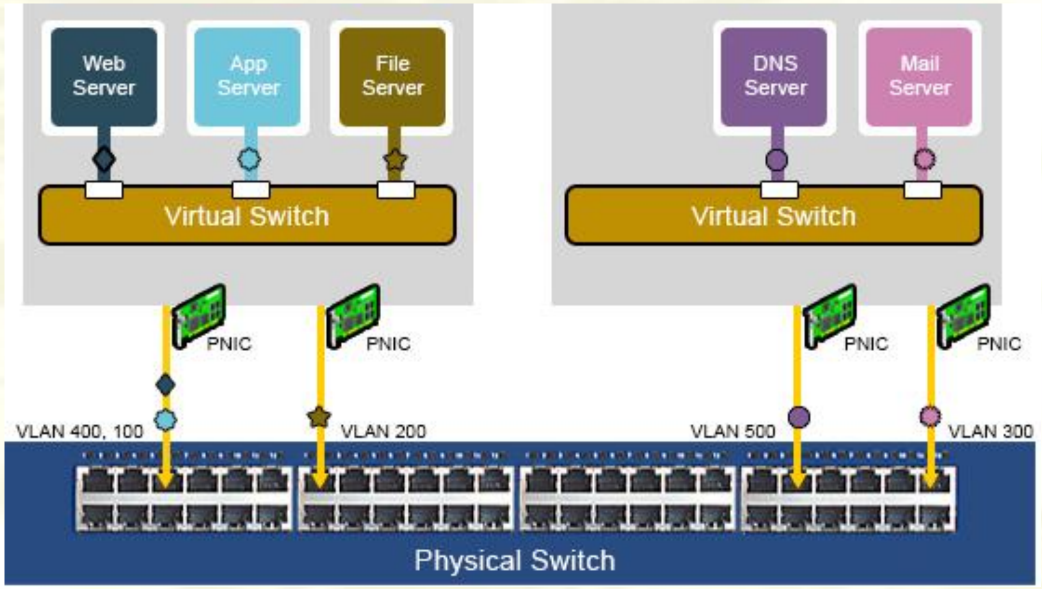
SDN and NFV are not in contrast and can coexist
SDN, more focused on optimizing network infrastructure
NFV, more focused on optimizing the network functions



- Optimize network infrastructure such as Ethernet switches, routers and wireless access points
- OSI Layer 2-3

- Optimize deployment of network functions such as: load balancer, firewall, WAN optimization controller, deep packet inspection etc.
- OSI Layer 4-7

Use Cases (i)

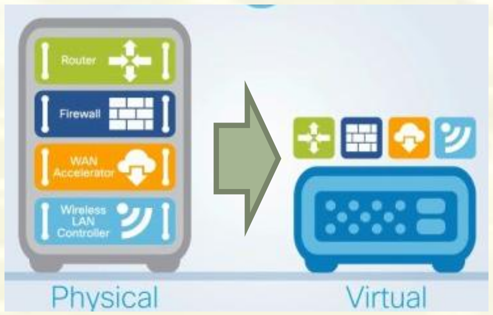


New networks and virtual network services can be instantiated as VMs are created or destroyed

Virtual Networks inside Cloud Computing platforms:
Network functions are virtualized by definitions as they are implemented on top of the cloud virtual infrastructure

The cloud orchestrator (the software controlling the instantiation of VMs) is already a NFV orchestrator

Use Cases (ii)



Home Routers or customer network equipment

Customer network equipment can be virtualized.

This allows the deployment of new services in the customer's network as the customer requires (and pay) them. New services (unknown at the time of device installation) can be created and deployed anytime.

Use Cases (iii)

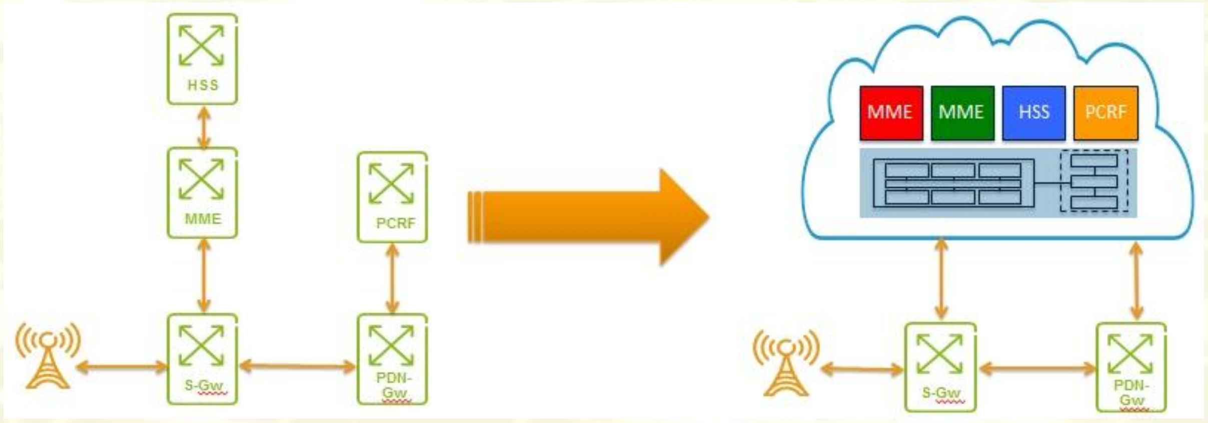
The core network of wireless operator is a complex infrastructure. Wireless protocols rapidly changes over time (e.g. UMTS -> LTE -> LTE Advanced -> 5G)

The virtualization of the core network can allow:

(i) Network reconfiguration and protocol update using the same hardware

(ii) Rapid deployment of virtualized network services to create Virtual Network Service Operators

Evolved Packet Core Networks (EPC) of Wireless Providers

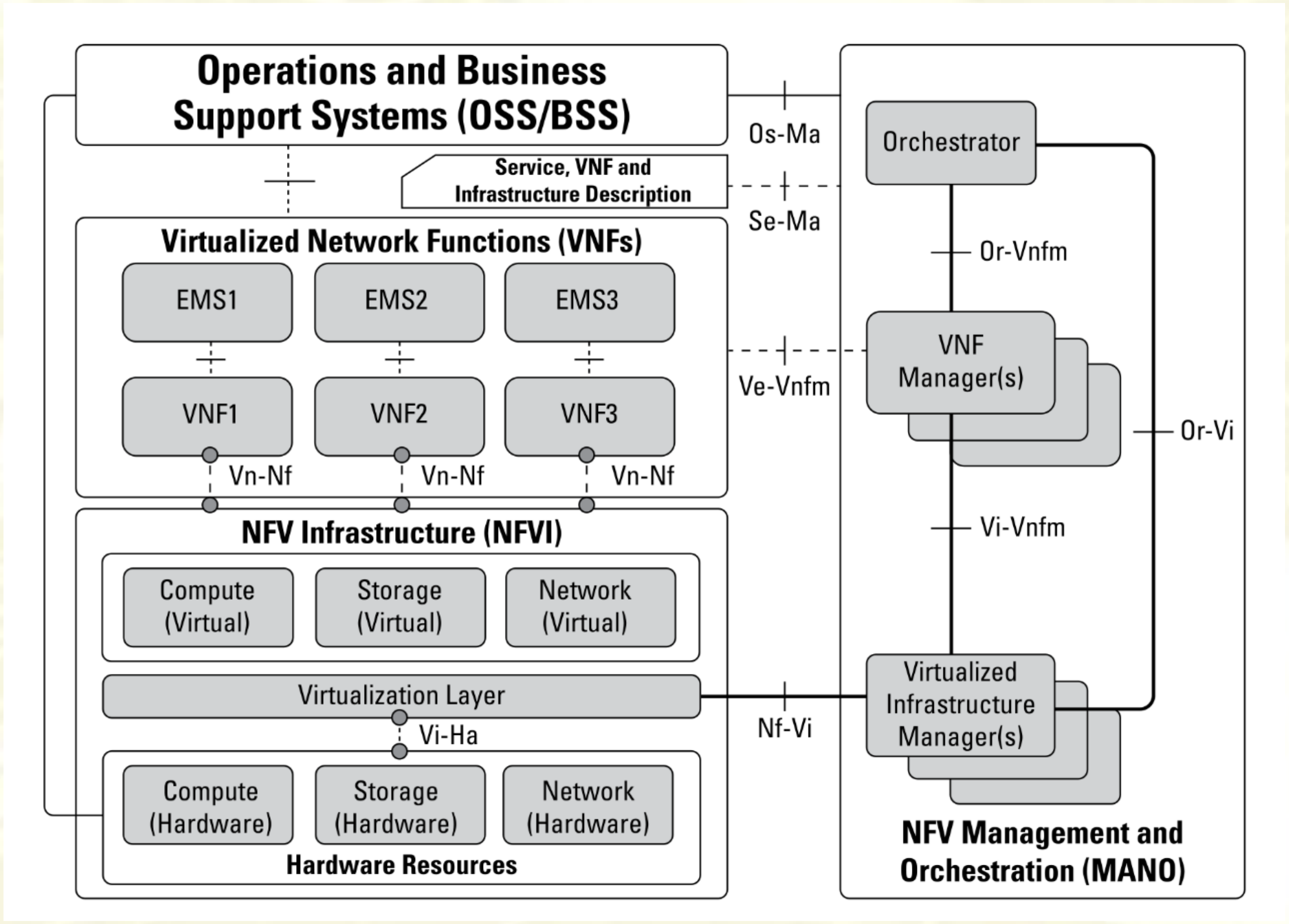


NFV - Advantages

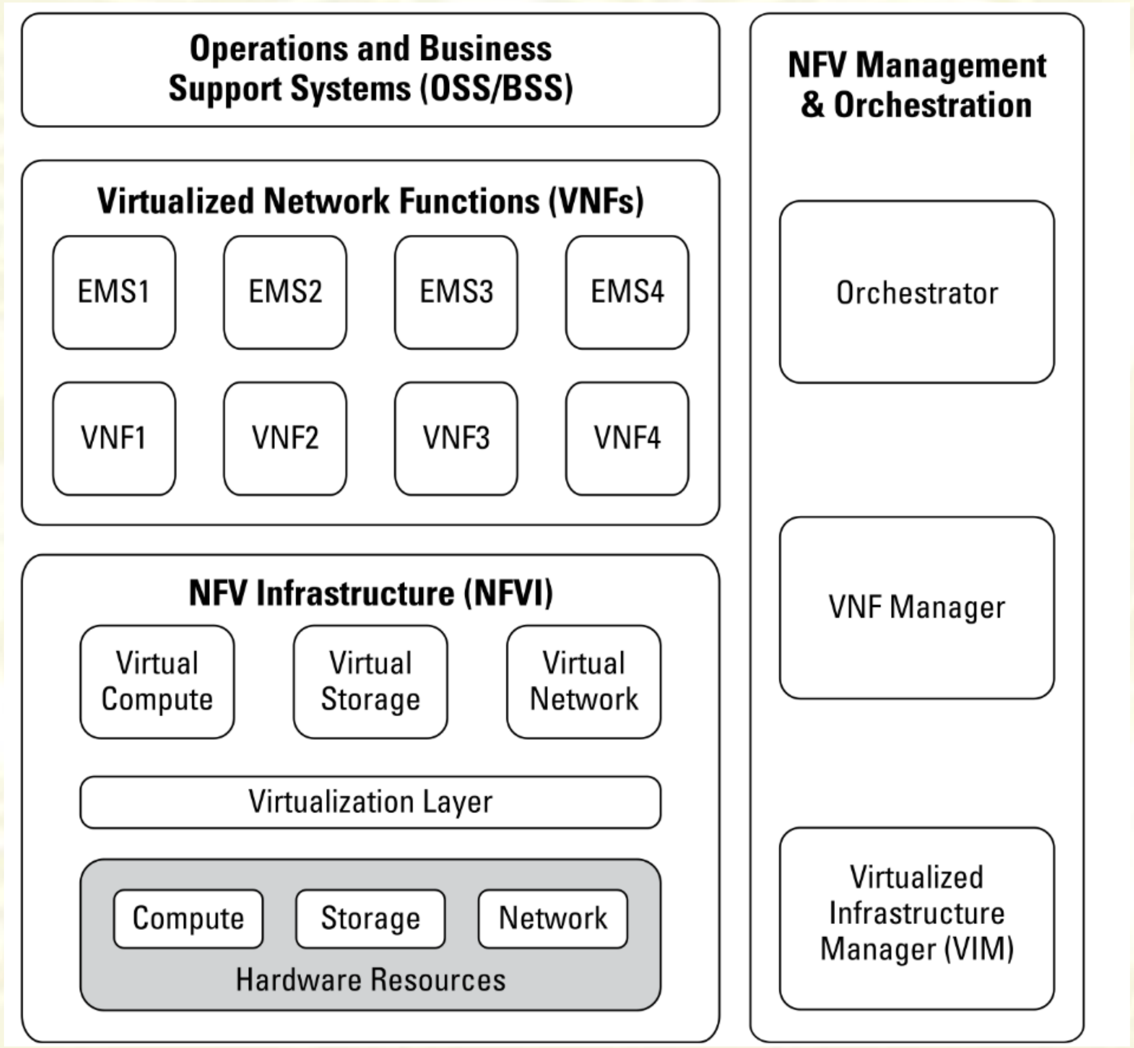
- It enables new opportunities and more innovation
  - The same hardware can be used to create new services, unknown at design time
- High flexibility
- Faster time to market for new services
- Improved business processes

Standard Architecture

-
The ETSI NFV Industry Specification Group in its working group MANO has defined a high level functional architectural framework for NFV



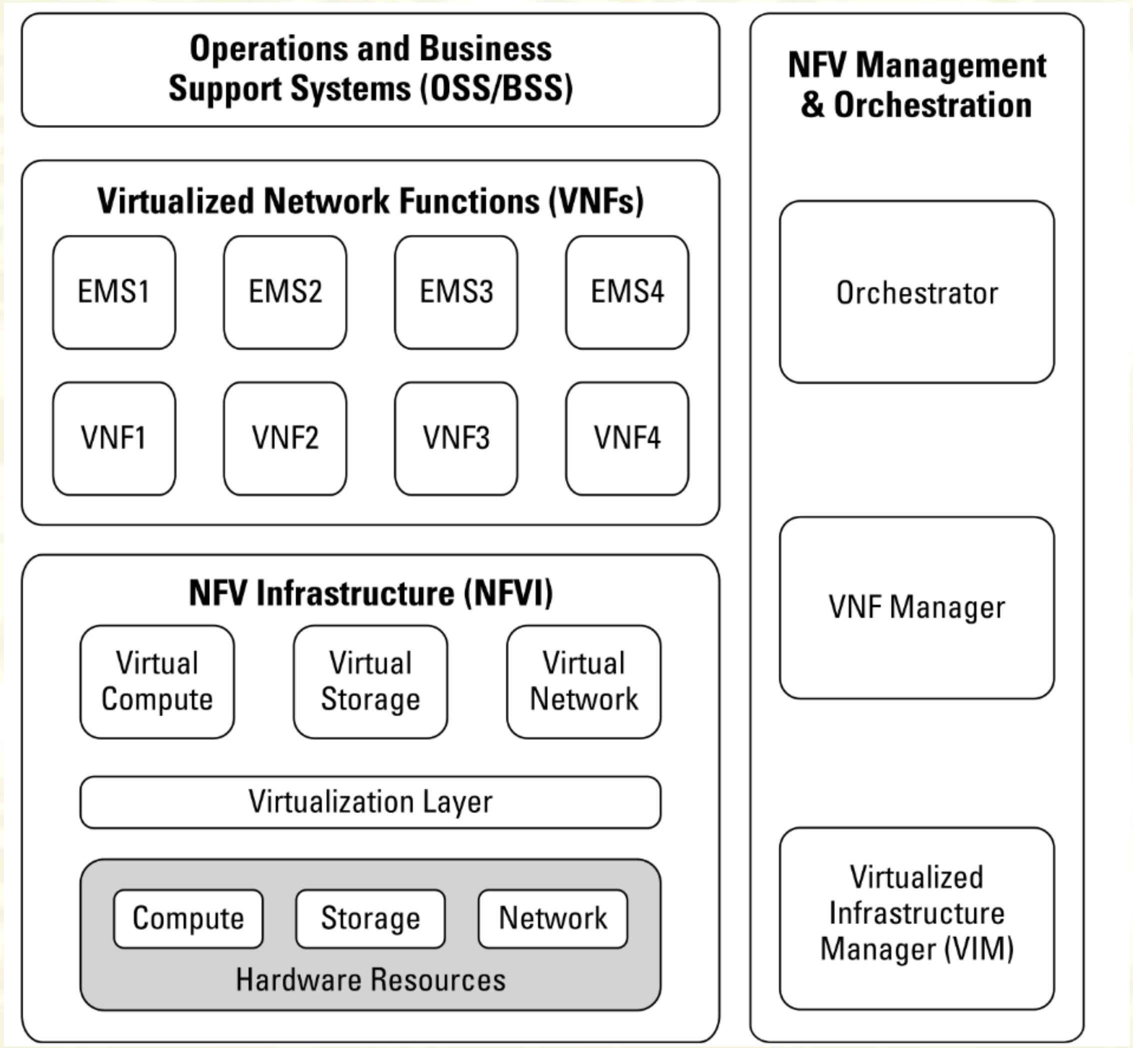
ETSI MANO


Operations and Business Support Systems (OSS/BSS)
Virtualized Network Functions (VNFs)
EMS1
EMS2
EMS3
EMS4
VNF1
VNF2
VNF3
VNF4
NFV Infrastructure (NFVI)
Virtual Compute
Virtual Storage
Virtual Network
Virtualization Layer
Compute
Storage
Network
Hardware Resources
NFV Management & Orchestration
Orchestrator
VNF Manager
Virtualized Infrastructure Manager (VIM)

Network Functions Virtualization Infrastructure (NFVI):
A subsystem that consists of all the hardware and software components *on top of which VNFs are deployed*.

One or more VNFs are instantiated to implement a **Network Service (NS)** to implement a specific network functionality.

ETSI MANO

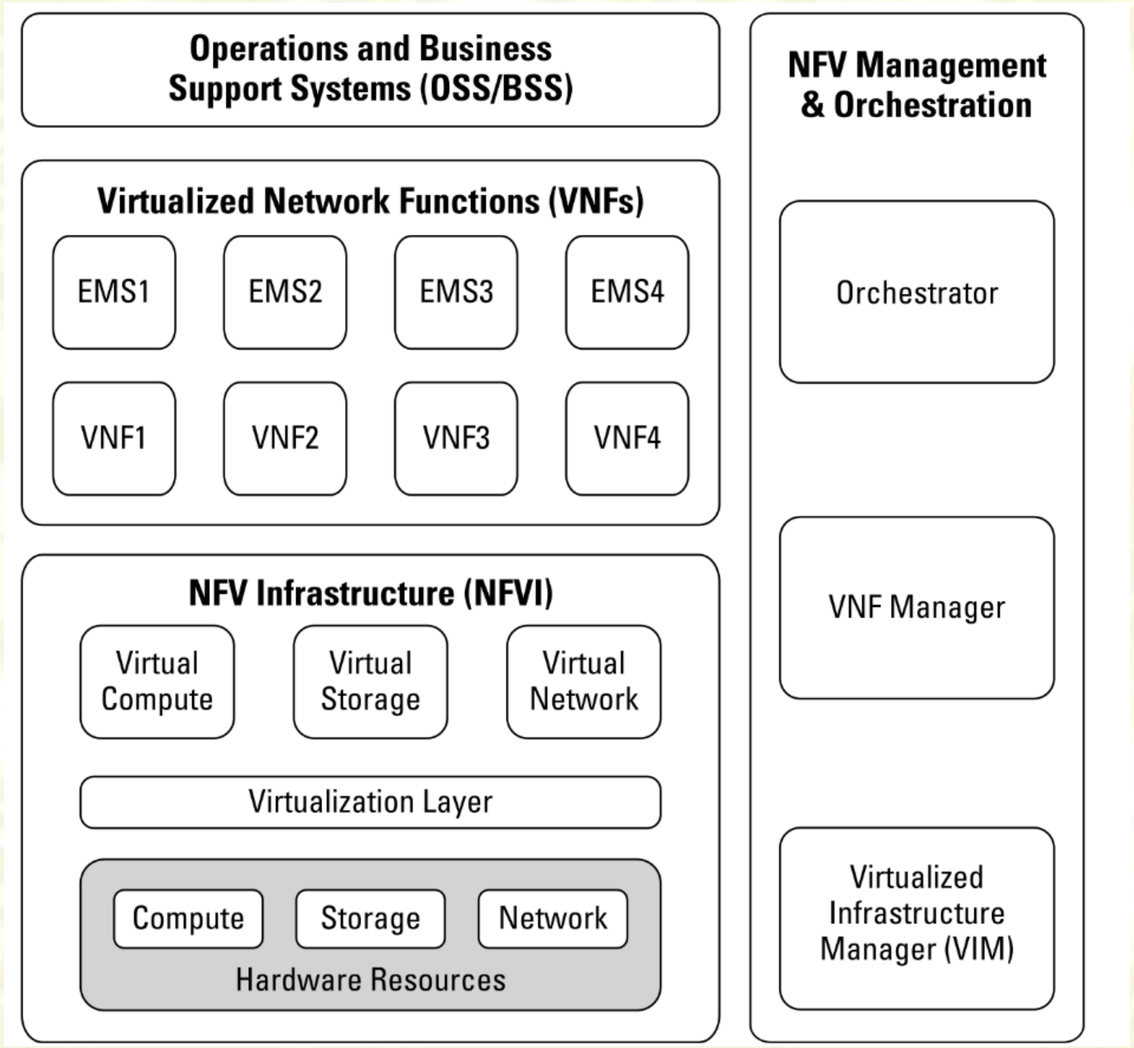


Operations and Business Support Systems (OSS/BSS)
NFV Management & Orchestration
Virtualized Network Functions (VNFs)
EMS1
EMS2
EMS3
EMS4
VNF1
VNF2
VNF3
VNF4
Orchestrator
NFV Infrastructure (NFVI)
Virtual Compute
Virtual Storage
Virtual Network
VNF Manager
Virtualization Layer
Compute
Storage
Network
Hardware Resources
Virtualized Infrastructure Manager (VIM)

Management and Orchestration (MANO):
A subsystem that includes the **Network Functions Virtualization Orchestrator** (NFVO), the **Virtualized Infrastructure Manager** (VIM), and the **Virtual Network Functions Manager** (VNFM).

VNFM: is responsible for the management and operation of individual VNFs
NFVO: is responsible for managing network services (NS) that span multiple VNFs

ETSI MANO



Operations and Business Support Systems (OSS/BSS)
Virtualized Network Functions (VNFs)
EMS1
EMS2
EMS3
EMS4
VNF1
VNF2
VNF3
VNF4
NFV Infrastructure (NFVI)
Virtual Compute
Virtual Storage
Virtual Network
Virtualization Layer
Compute
Storage
Network
Hardware Resources
NFV Management & Orchestration
Orchestrator
VNF Manager
Virtualized Infrastructure Manager (VIM)

VIM: is responsible for managing the physical and virtual resources. It interfaces with the NFV Infrastructure (possibly across different hosts).

It manages the resources based on the input from the VNFM and NFVO

References

- http://www.ttcenter.ir/ArticleFiles/ENARTICLE/3431.pdf
- "Network Function Virtualization for dummies": https://h20195.www2.hpe.com/V2/getpdf.aspx/4AA6-6386ENW.pdf

Docker Refresher

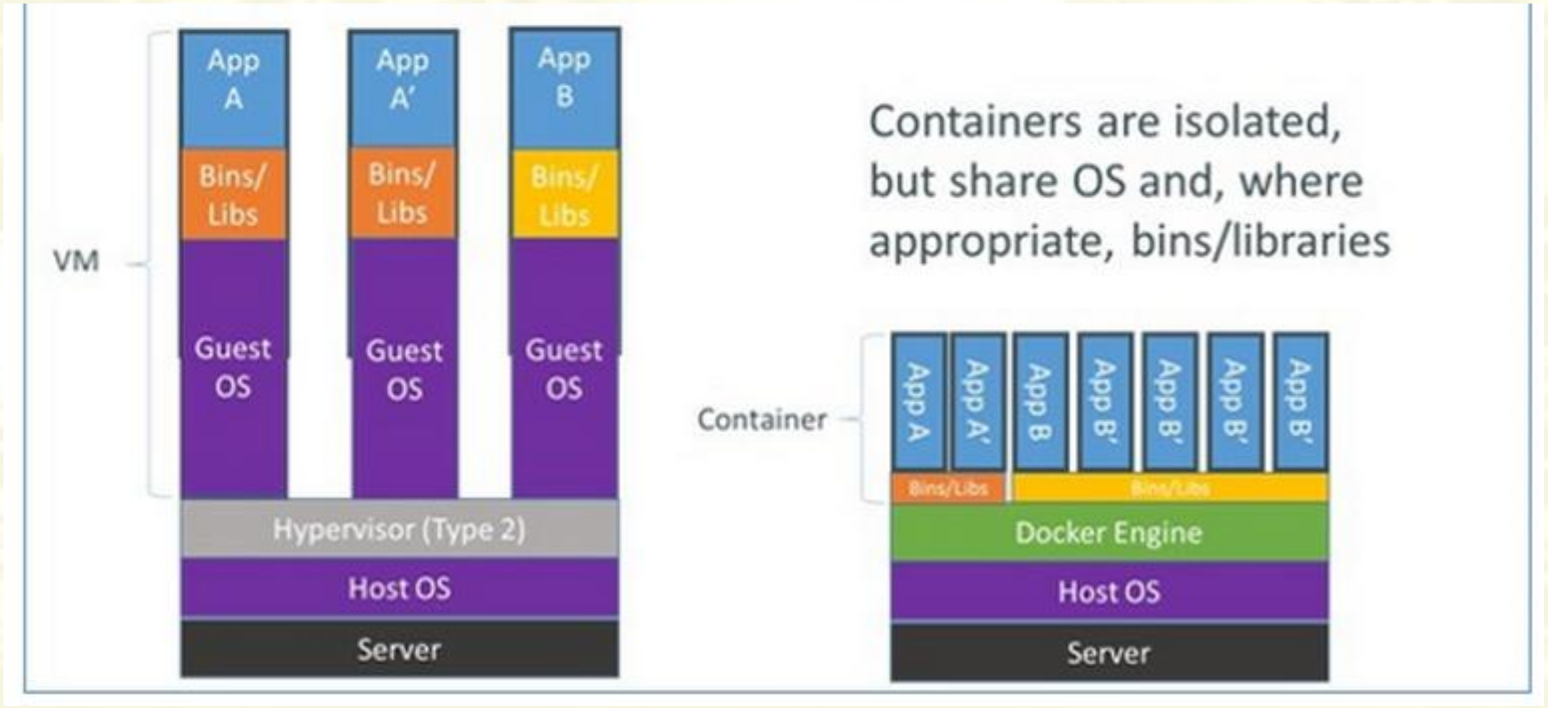
Carlo Vallati
Assistant Professor@ University of Pisa
c.vallati@iet.unipi.it

NFV – Virtualization Technologies

- Many virtualization technologies can be used
- In cloud computing platforms, the virtual cloud network infrastructure is used: OpenStack, AWS, Microsoft Azure, etc
  - VNFs are implemented as virtual machines
- In other contexts, other virtualization technologies can be adopted
- **Docker** is a lightweight virtualization technology based on ***containers***

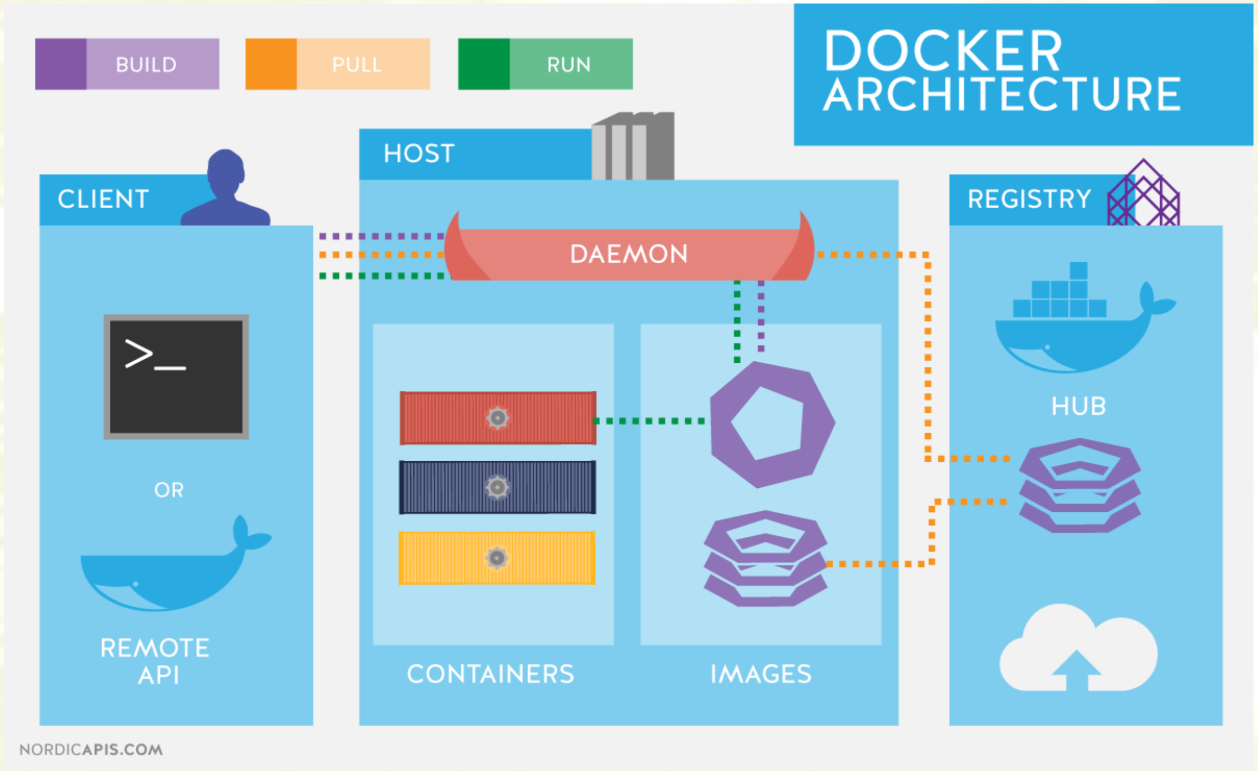
Docker

Docker is a container technology for Linux that allows a developer to package up an application with all of the parts it needs. Containerization is an operating system-level virtualization. Containers share the same kernel of the Host OS, which implements a set of functions to guarantee isolation.



Docker

Containers are created from locally available images. Images are created from standard images that can be downloaded from public repositories. All operations are managed by the Docker daemon.



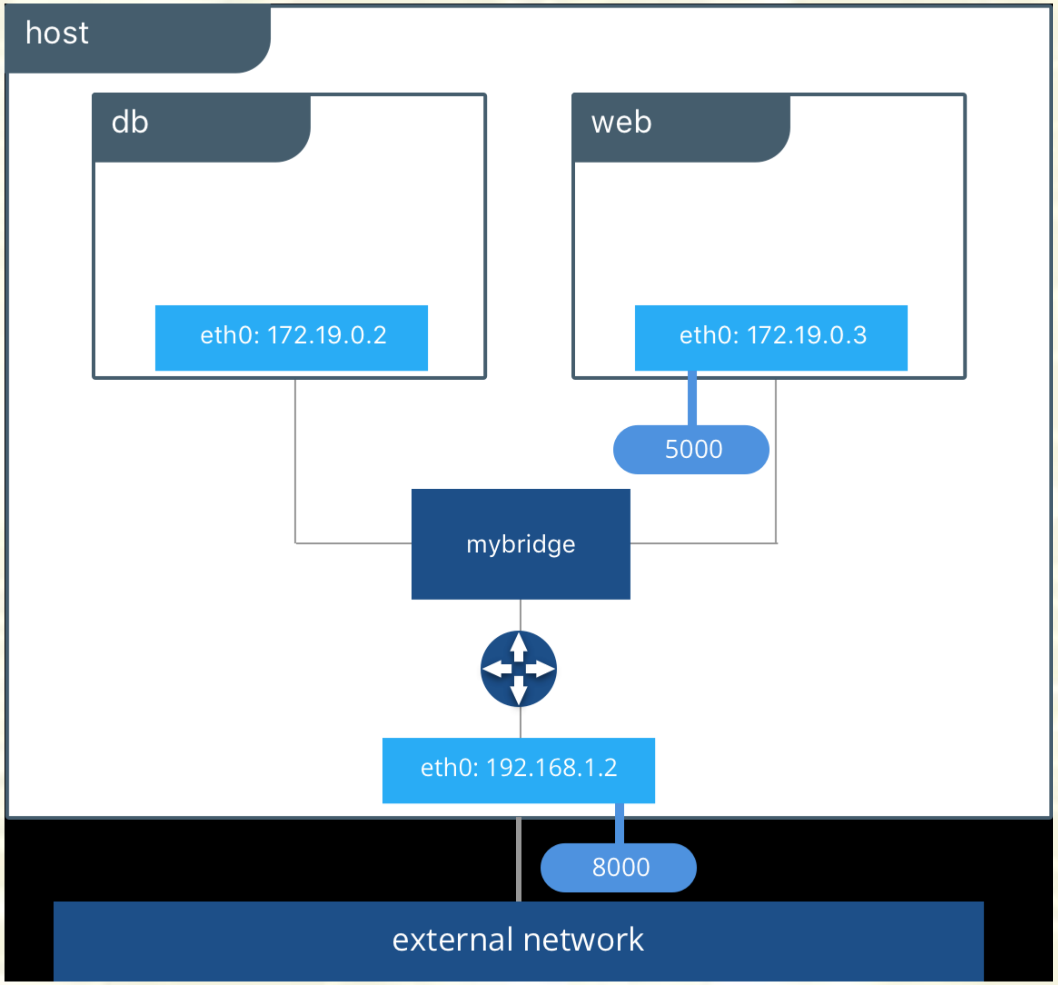
Docker Networking

The Docker engine creates a virtual network to allow communication among different containers.

In the default mode, for every container a virtual network adapter is created

Every virtual interface have a private IPv4 address, communication among different containers can happen through routing or trough L2 forwarding if interfaces a bridged (as in the picture).

A service listening on one port can be published to the external network, by using port forwarding



Create a new image

Create a new Docker image from one standard image from the repository:

1. Create a file Dockerfile, which describes how the new image has to be created, e.g.:

```
# Start from an official image with a Telnet server
FROM rohan/ascii-telnet-server
# Flag that the software inside this image listens on port 23
EXPOSE 23
# Specify the command to be run inside the container when it’s started
CMD ["python", "./chess.py"] (not needed in this case)
```

2. Build the image:

```
sudo docker build .
```

3. Check that the image is available:

```
sudo docker ps
```

Deploy and Check

Deploy a new container from one image locally available and check its status:

1. Deploy the container:

```
sudo docker run --name NAME_CONTAINER -d {-p
CONTAINER_PORT:PUBLIC_PORT} NAME_IMAGE
```

   e.g.

```
docker run --name telnet -d -p 23:23 rohan/ascii-telnet-
server
```

2. Check if the image is running:

```
sudo docker ps
```

3. Check the resources:

```
sudo docker stats
```

4. Check the port mapping:

```
sudo docker port CONTAINER_ID
```

5. Connect to the server:

```
telnet localhost
```

Docker Compose

Docker compose is a tool for defining and running multi-container applications. The deployment is described through a ‘yml’ configuration file, which includes the list of containers to deploy.

```
services:
  service3:
    image: NAME_IMAGE
    depends_on:
      - service1
      - service2
    restart: always
    environment:
      - SYSTEM_VARIABLE=value
ports:
      - "8080:8080"
```

1. Deploy the application:

```
sudo docker-compose -f file.yml create
```

2. Start the application:

```
sudo docker-compose -f file.yml up
```