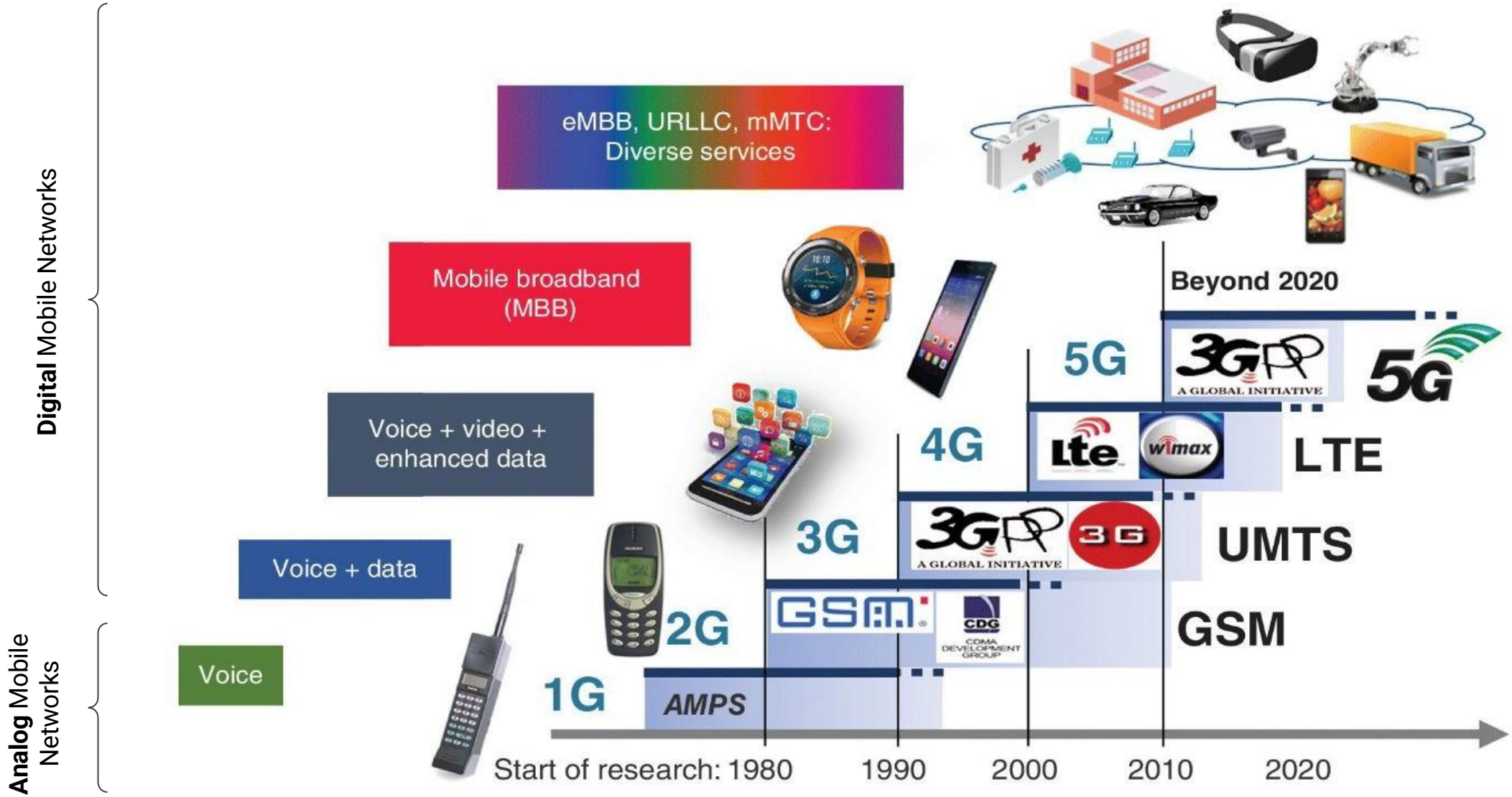
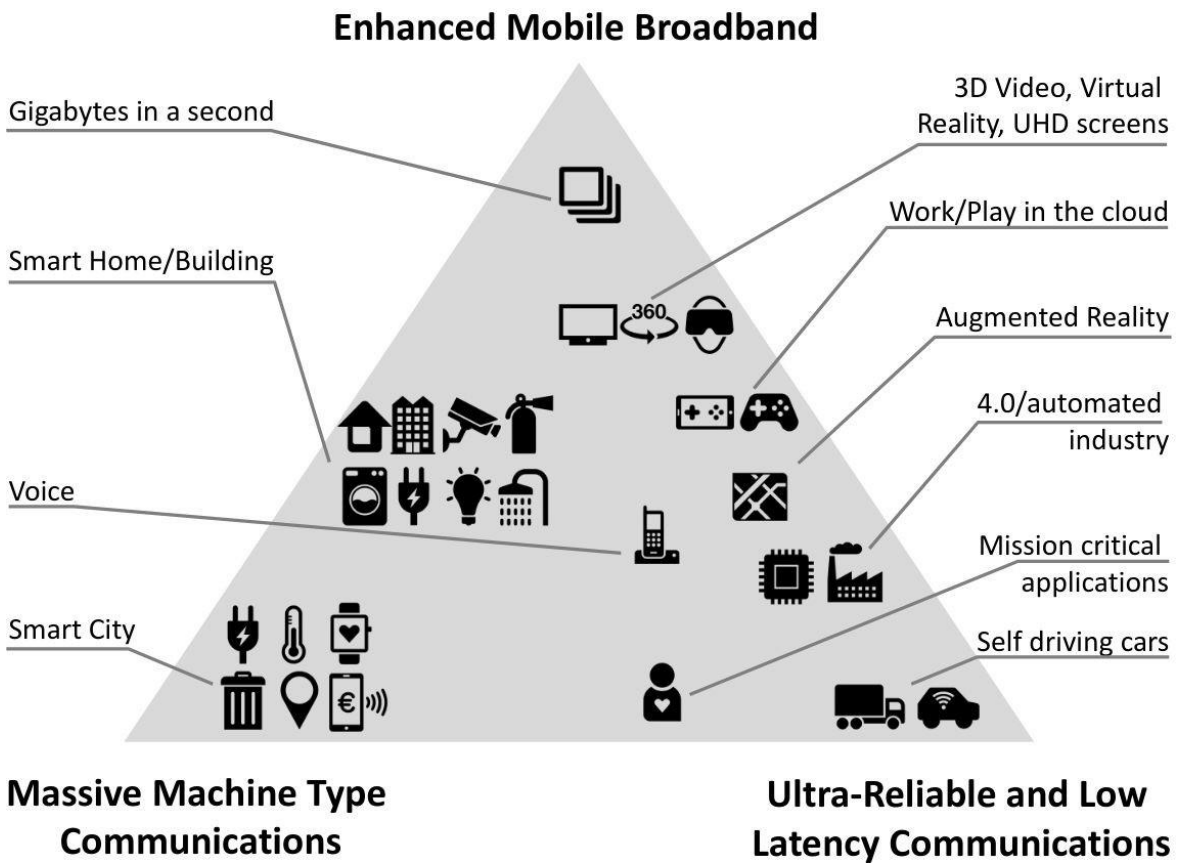
Last Lesson Recaps

Main drivers behind cellular communications



Main drivers behind cellular communications



[9]

Three Key Application Areas:

- **Enhanced Mobile Broadband (eMBB):** Better mobile phones and hot spots. High data rates, high user density. *Human centric communications*
- **Ultra-Reliable and Low-Latency Communications (URLLC):** Vehicle-to-Vehicle communication, mission critical communication, Industrial IoT. *Human and Machine centric communication*
- **Massive Machine Time Communications (mMTC):** Very large number of devices, low data rate, low power. IoT with long battery life time for smart cities. *Machine-centric communication*

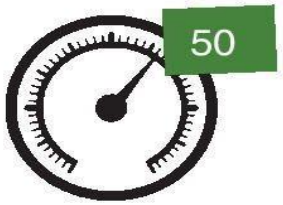
NGMN use case groups

Broadband access in dense areas

PERVASIVE VIDEO

Broadband access everywhere

50+ MBPS EVERYWHERE



Higher user mobility

HIGH SPEED TRAIN

Massive Internet of things

SENSOR NETWORKS

Extreme real-time communications

TACTILE INTERNET

Lifeline communications

NATURAL DISASTER

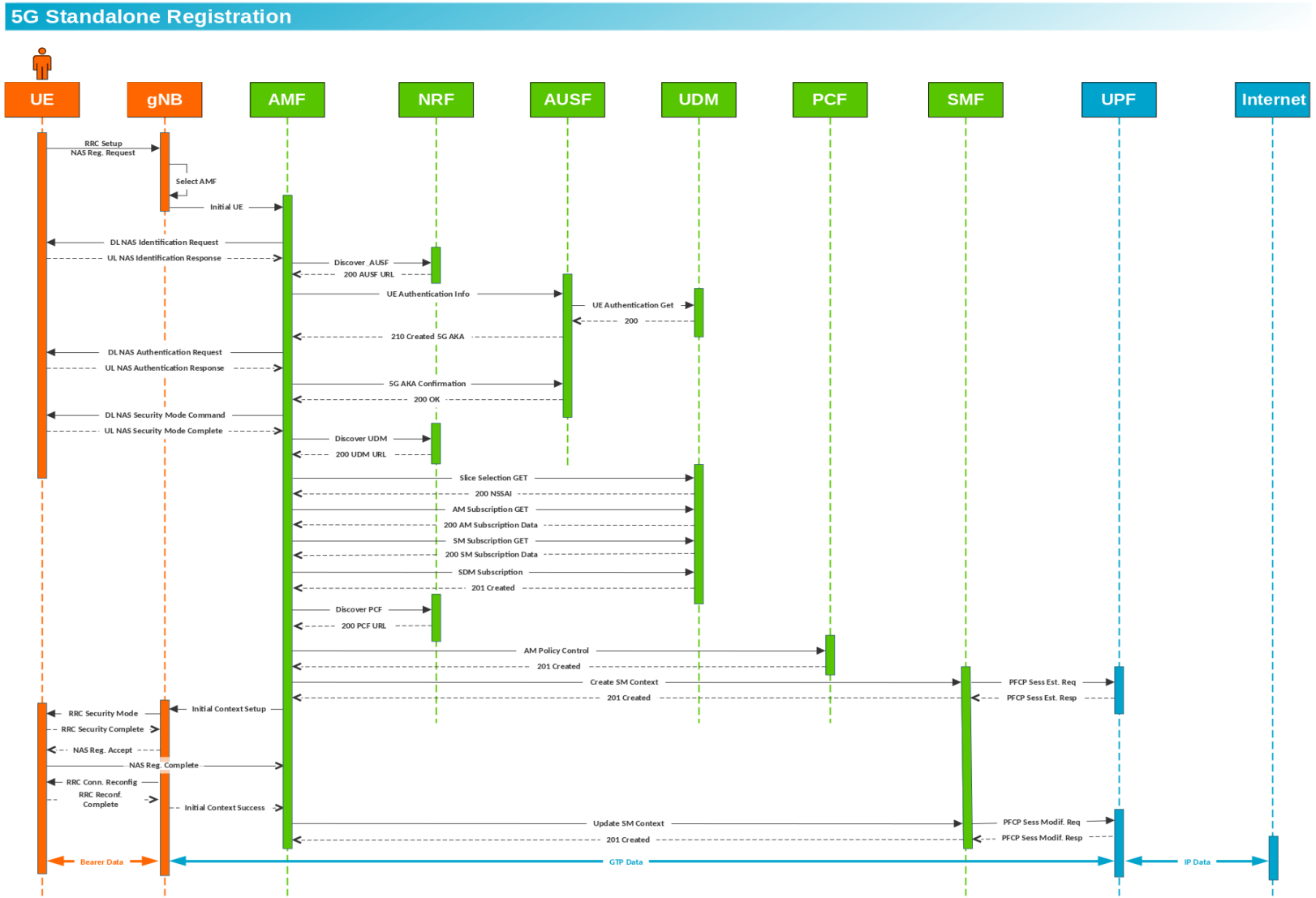
Ultra-reliable communications

E-HEALTH SERVICES

Broadcast-like services

BROADCAST SERVICES

UE Registration in 5G



Purposes of Registration?

- **Initial Access:**
  Enables UEs to connect to the 5G network for the first time.

- **Mobility Management:**
  Updates network information as the UE moves between Tracking Areas (TAs).

- **Capability Updates:**
  Notifies the network of any changes in the UE's capabilities or supported protocols.

- **Periodic Updates:**
  Maintains the connection with the network during inactive periods.

Message Contents of Registration Request (UE message):

Parameter	Role
PLMN ID	Identifies the specific Public Land Mobile Network the UE wishes to connect to.
SUCI (Subscription Concealed Identifier)	Provides a secure identifier while concealing actual subscription details for privacy.
5G-GUTI (Globally Unique Temporary UE Identity)	Offers a temporary identity to maintain privacy and security during communication.
Registration Type	Specifies the nature of the registration (initial, mobility, periodic), guiding network processing.
Security Parameters	Ensures secure authentication and integrity protection for communication.
Requested NSSAI (Network Slice Selection Assistance Information)	Informs the network about the specific network slice required for tailored services.
UE Capabilities	Communicates the UE's radio capabilities and supported features for compatibility.
PDU Session Status	Provides information about existing data connections for effective resource management.
Other Parameters	Includes additional information DRX power saving features to optimize performance and resource usage.

UE Registration steps:

Step1: The UE begins communication with the network, indicating its desire to register and access services (using UE Message).

Step2: The RAN chooses the AMF responsible for handling the UE's registration. The RAN uses the UE's location, requested NSSAI, and operator policies to determine the appropriate AMF.

Step3: The RAN sends the UE's registration request to the selected AMF over the N2 interface (UE message+cell ID+user location).

Step4 (optional): If the serving AMF has changed, this step ensures the new AMF gets all the necessary information about the UE from the old AMF.

Step5: The AMF chooses an AUSF (consulting the NRF) to perform authentication of the UE. **Selection Criteria:** Based on the UE's SUCI.

Step6: The AUSF uses the UE's credentials and security information from the UDM (Unified Data Management) to authenticate the UE.

Step7: The AMF selects a Unified Data Management (UDM) (consulting the NRF) function to access subscriber-related information (SM, SDM, AM subscription).

Step8: AM policy control with PCF

Step9: AMF sends a create SM context to SMF

Step10: The SMF establish the session with UPF

Step11: UE access to data

1. Access Management (AM)

Role: Manages user access to the network.
Example Content:
Authentication Request:
User ID: IMSI1234567890
Authentication Method: 5G-AKA
Access Control:
Allowed Services: Voice, Data
Denied Services: Roaming

2. Session Management (SM)

Role: Manages data sessions for users.
Example Content:
- **Session Establishment Request**:
  - **User ID**: IMSI1234567890
  - **Requested QoS**: High
  - **APN (Access Point Name):** internet
- **Session Modification**:
  - **Change in QoS**: Medium to High
  - **Additional Resources**: 5GB Data

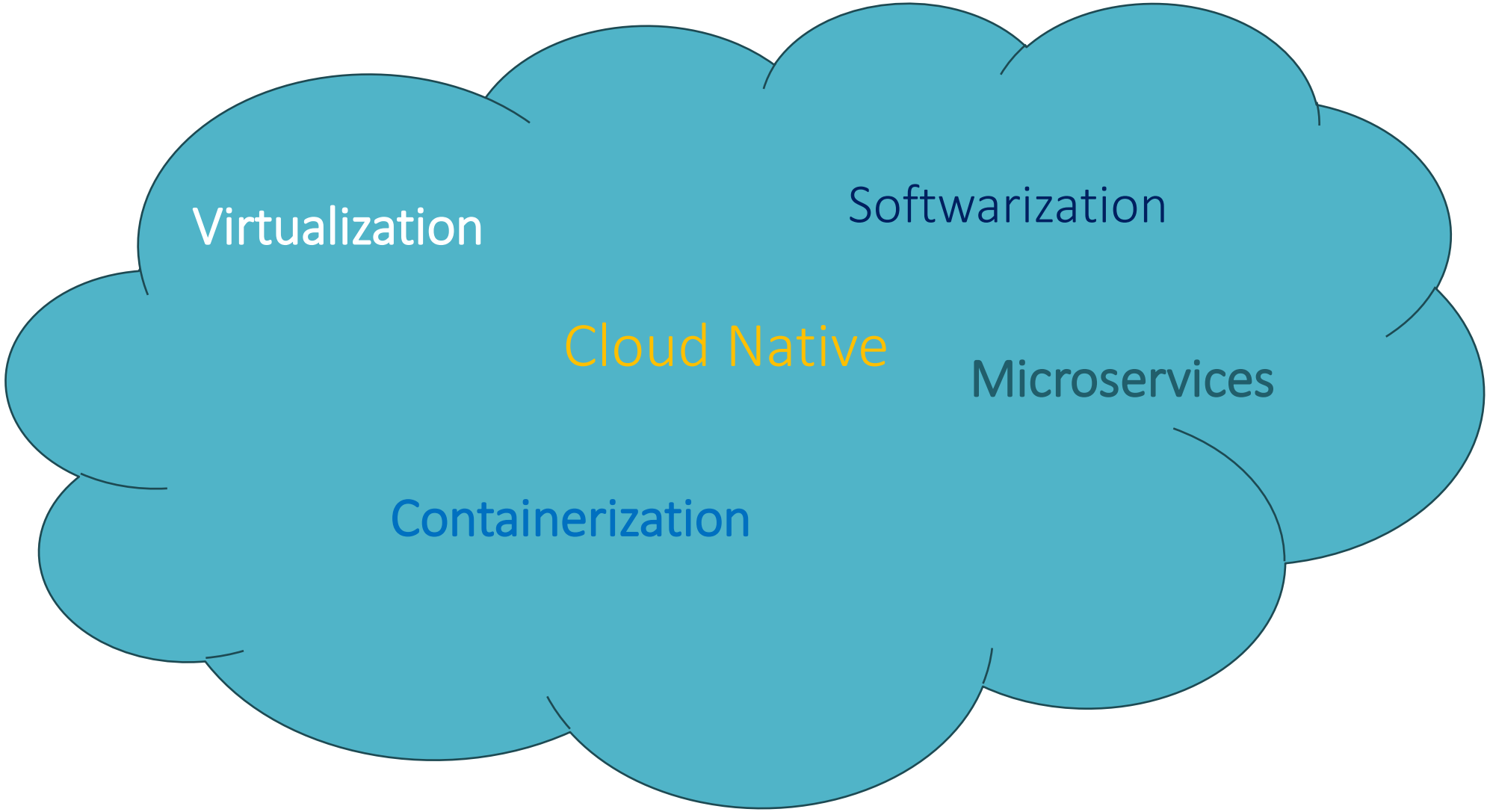
3. **Subscription Data Management (SDM)**

Role: Manages subscriber data and profiles.
Example Content:
- **Subscriber Profile**:
  - **User ID**: IMSI1234567890
  - **Subscription Type**: Postpaid
  - **Data Limit**: 20GB/month
- **Service Preferences**:
  - **Preferred Language**: French
  - **Roaming Settings**: Enabled

What is Virtualization ?

2. Virtualization Concepts

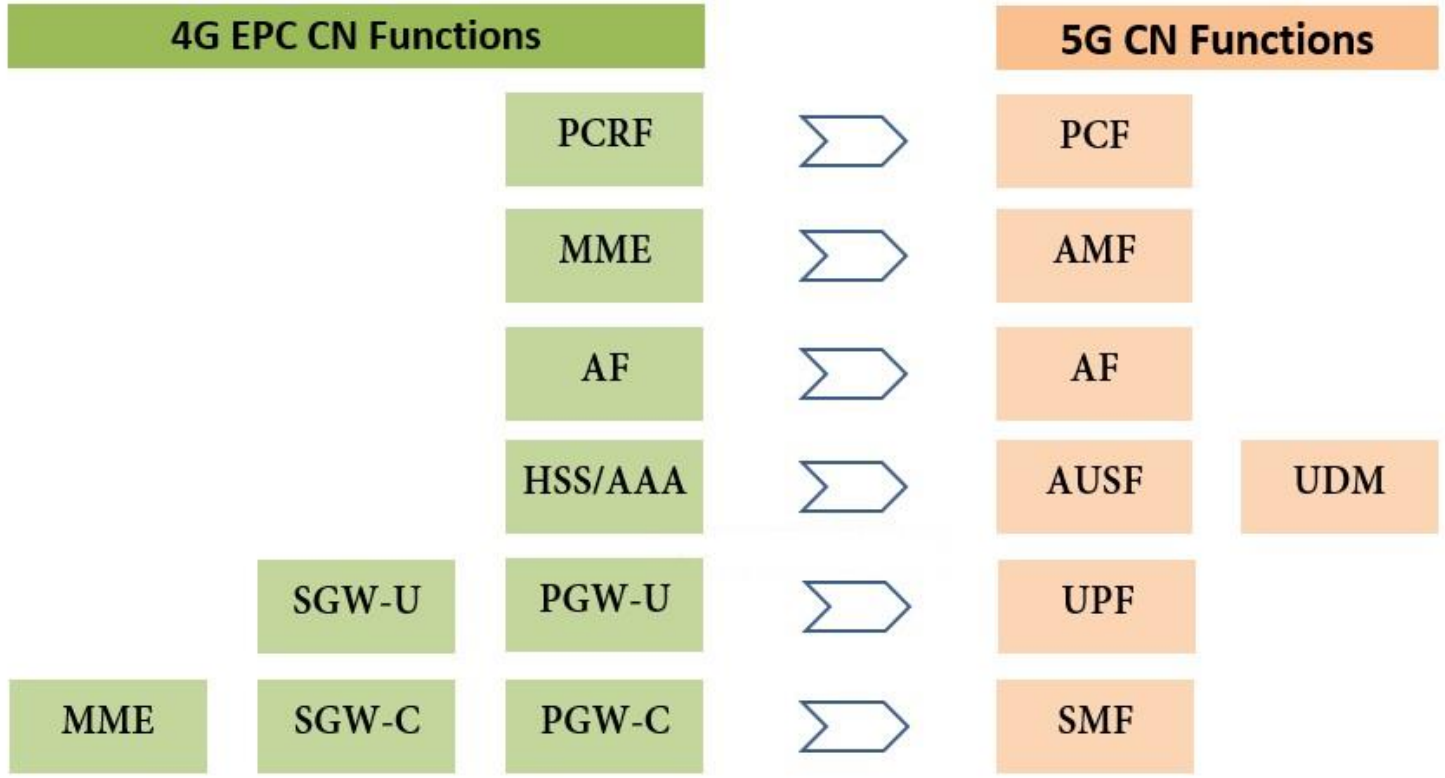
- Where does it come from ?
- Basics of virtualization
- Virtualization Techniques
- Virtualization Benefits
- From Server to Network Virtualization


Virtualization
Softwarization
Cloud Native
Microservices
Containerization

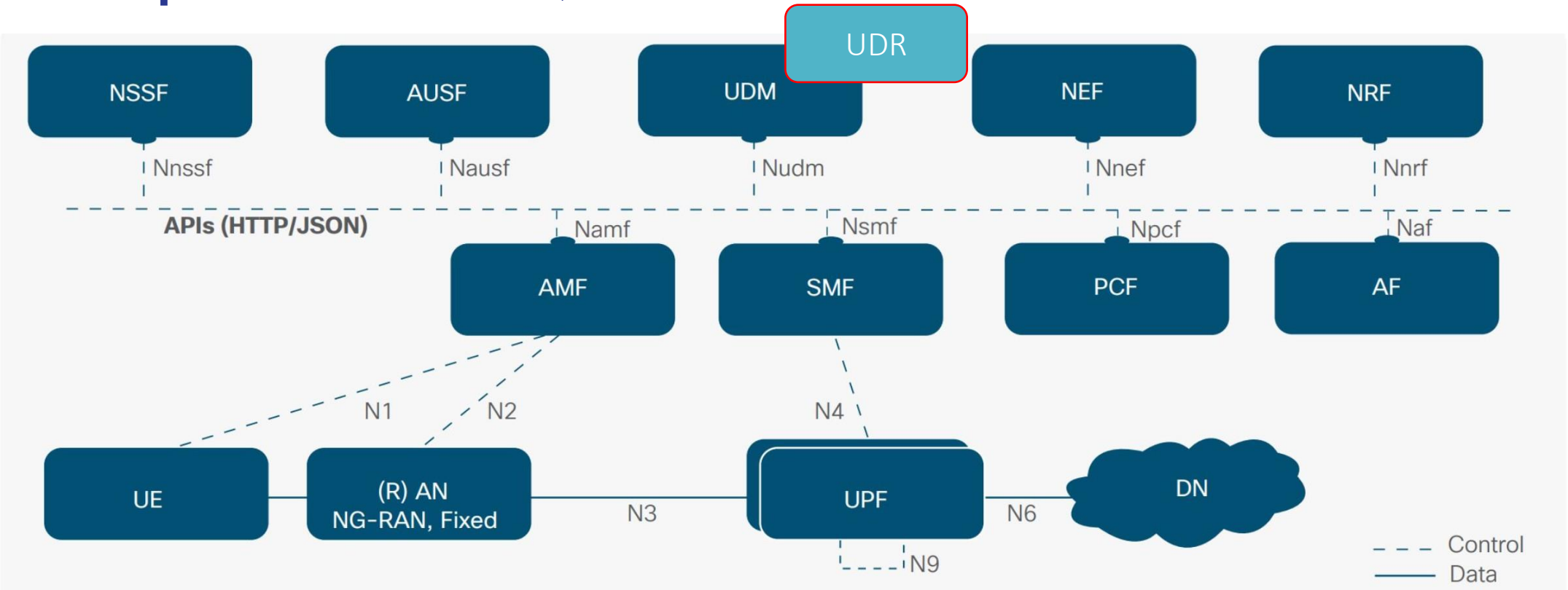
2. Virtualization Concepts

Core Networks Evolution from 4G to 5G

4G vs 5G Functions:



Compared to 4G, 5G has more functions



NSSF	Network Slice Selection Function
AUSF	Authentication Server Function
UDM	Unified Data Management Function
NEF	Network Exposure Function
NRF	Network Repository Function

AMF	Access and Mobility Management Function
SMF	Session Management Function
PCF	Policy Control Function
AF	Application Function
UDR	Unified Data Repository

UE	User Equipment
RAN	Radio Access Network
UPF	User Plane Function
DN	Data Network

5G system Service Based Architecture (SBA)

Service Based Architecture (SBA) a.k.a. Service Oriented Architecture (SOA)

- **Standardized:** Services within the same inventory (e.g. 5GC) are in compliance with the same contract design standards and have a standard communication agreement
- **Loosely coupled:** Inter-service contracts (e.g. NFs) dependency is minimized to the level that they are only aware of their existence. Interoperability between them is guaranteed
- **Scalability:** Network functions are broken down into independent services, allowing for better scalability
- **Reusable:** Resources, logic and functionalities are decomposed into multiple services to maximise code reusability
- **Composable:** Services can be composed to create new services (e.g. UPF chaining)
- **Autonomic:** Services have the control over their resources and functionalities. Minimal (or zero) external dependencies
- **Stateless:** Service do not store state for too long (or at all) of a transaction. Requests are treated independently
- **Abstract:** Inner logic is completely transparent to the consumer.
- **Discoverable:** Services can be found and identified

In the end of this course you will be able to answer this questions:

What is virtualisation?

The virtualisation landscape?

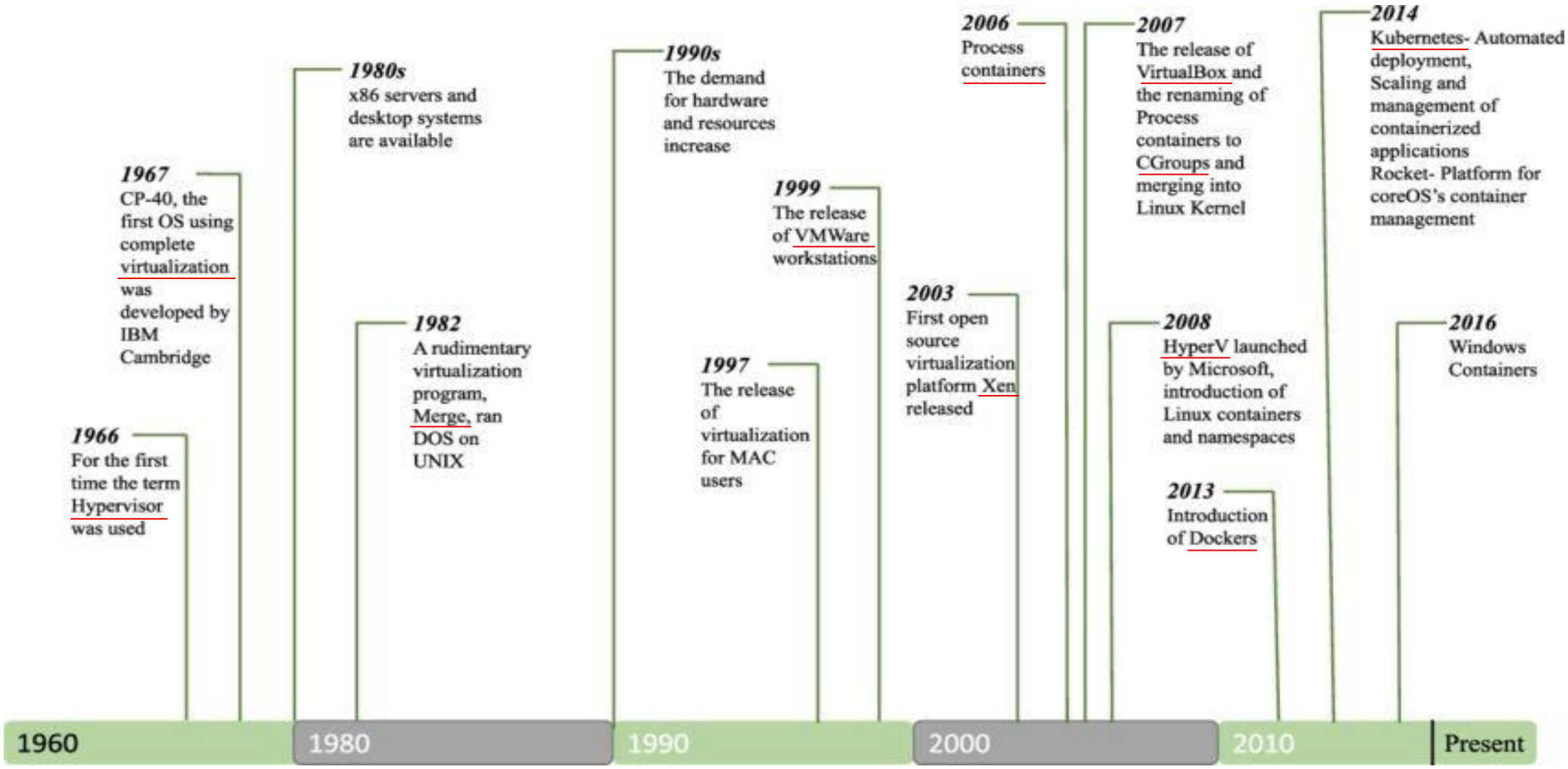
What is a container?

What is the difference between a VM and container?

Explain the concept of microservices?

The benefits of containerization of functions?

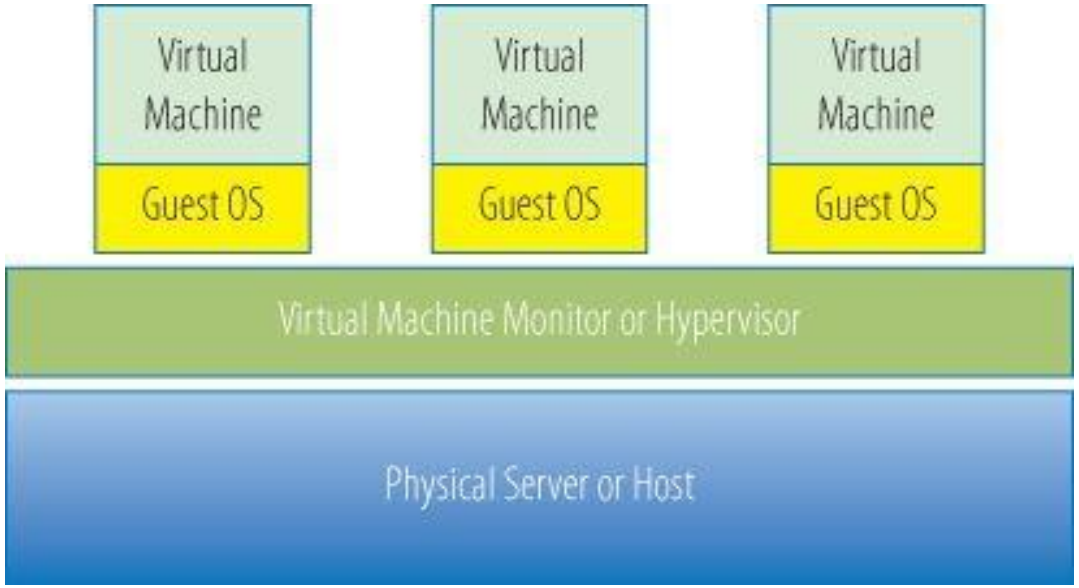
Looking back at 60 years of virtualization history



Definitions

- **Virtual Machine (VM)** is a software that can virtualize hardware resources, such as: processors, memory, storage, and network connectivity

- **Virtual Machine Monitor (VMM)**, which today is commonly called a **hypervisor**, is the software that provides the environment in which the VMs operate [11]



Definitions

Virtualization

- Virtualization is the abstraction of the true underlying hardware or software from VM Operating System (OS), application, storage or network

- Since we cannot run two operating systems on the same hardware at the same time (OSs like to have the full control !), virtualization makes an OS believe it's running on its own physical machine and has the full control of it

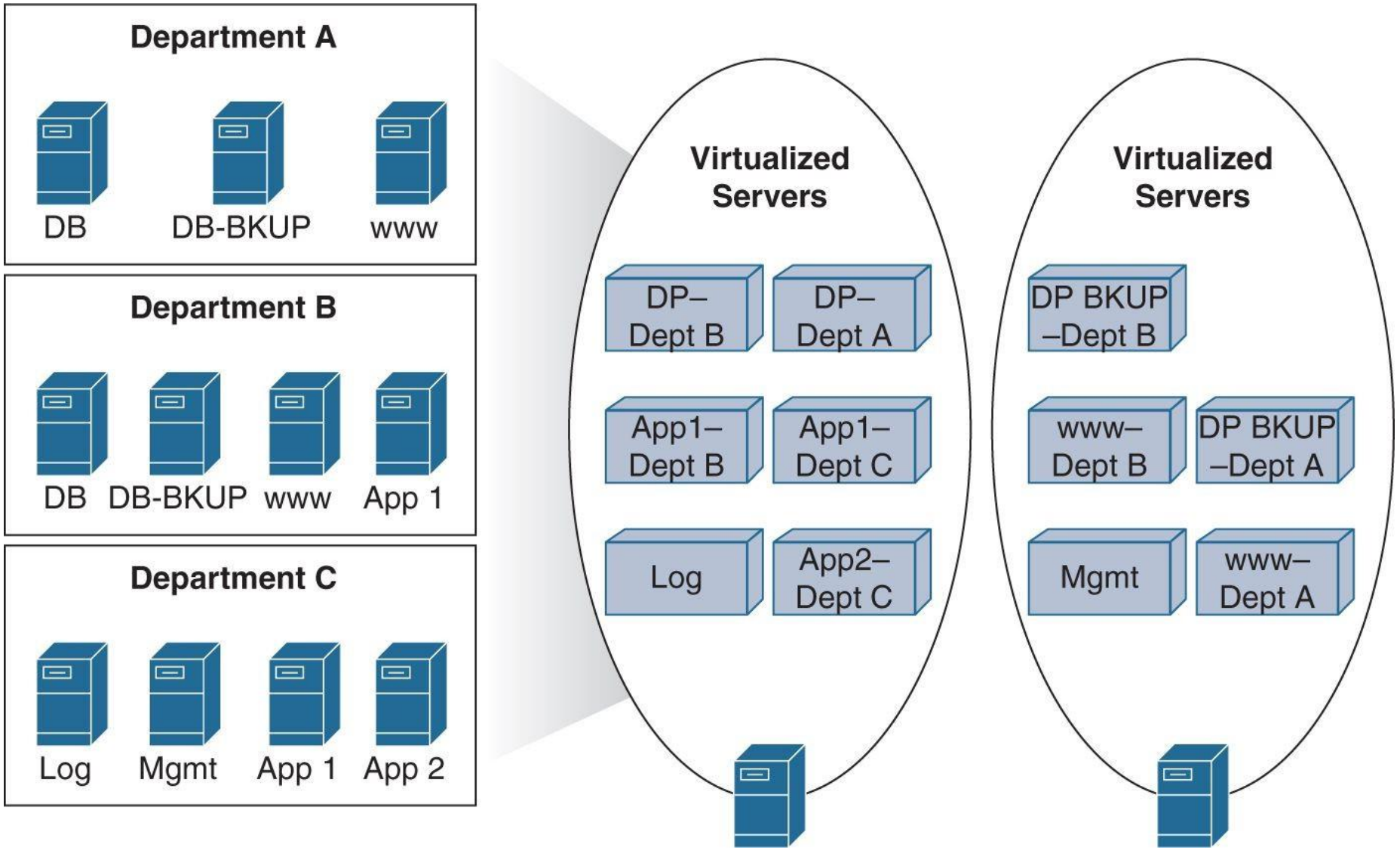
  VM OS is called **Guest OS**
  Physical machine OS is called **Host OS**

Definitions

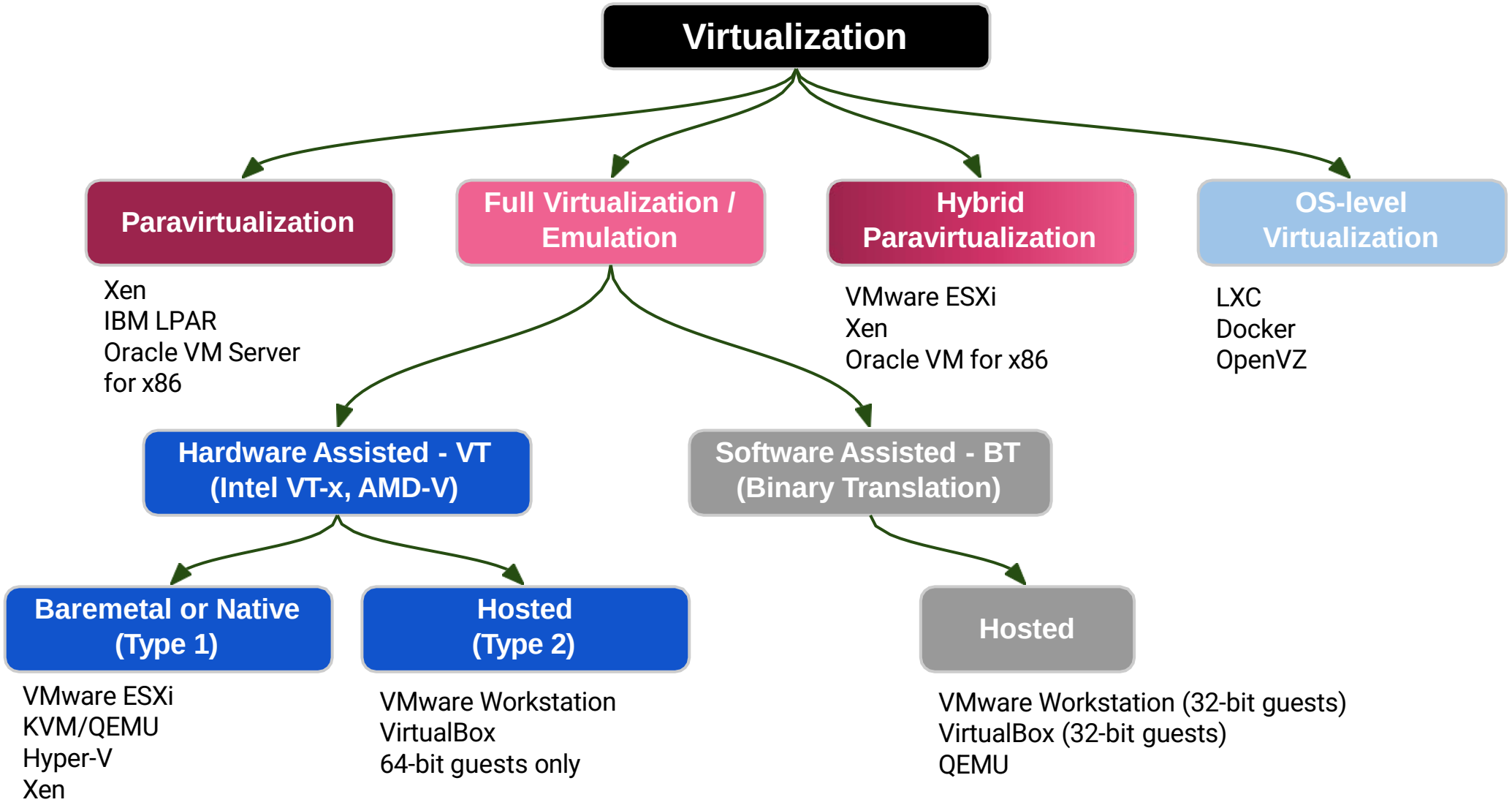
VMM/hypervisor properties:

- **Fidelity:** the environment it creates for the VM is essentially identical to the original (hardware) physical machine
- **Isolation or Safety:** the VMM must have complete control of the system resources
- ***Performance*:** there should be little or no difference in performance between the VM and a physical equivalent. It heavily depends on the virtualization technology

Server Virtualization



Virtualization Landscape



Definitions

1.Full Virtualization:

The hypervisor (virtual machine monitor) provides a complete simulation of the underlying hardware, allowing unmodified guest operating systems to run. Examples include VMware and Microsoft Hyper-V:

1.1 Hardware-Assisted Virtualization VT:

Modern CPUs (like Intel VT-x and AMD-V) include hardware support for virtualization, improving performance by allowing the hypervisor to manage resources more efficiently.

1.2 Software assisted Virtualization: using Binary Translation (BT) is a technique that allows unmodified guest operating systems to run on a hypervisor by translating their instructions into a format that the host CPU can execute.

2.Paravirtualization:

The guest OS is modified to be aware of the hypervisor, allowing for more efficient communication and resource management. Examples include Xen and some implementations of KVM.

5. Hybrid paravirtualization: is a virtualization technique that combines aspects of both paravirtualization and hardware-assisted virtualization.

6. Containerization (Os-level):

While not traditional virtualization, containerization (e.g., Docker) allows multiple isolated applications to run on a single OS kernel, sharing resources while maintaining separation.

Virtualisation:

- CPU virtualization
- Network virtualization
- Storage virtualization
- ...

X86 architecture

- The x86 architecture originated with the Intel 8086 microprocessor, released in 1978.
- It was the first 16-bit processor in the x86 family.
- x86 includes a rich set of instructions for various operations: arithmetic, logic, control flow, and data manipulation
- Intel expanded x86 to 32-bit (known simply as x86) and later to 64-bit (known as x86-64 or AMD64), significantly increasing memory addressing capabilities.

X86 architecture levels of privileges

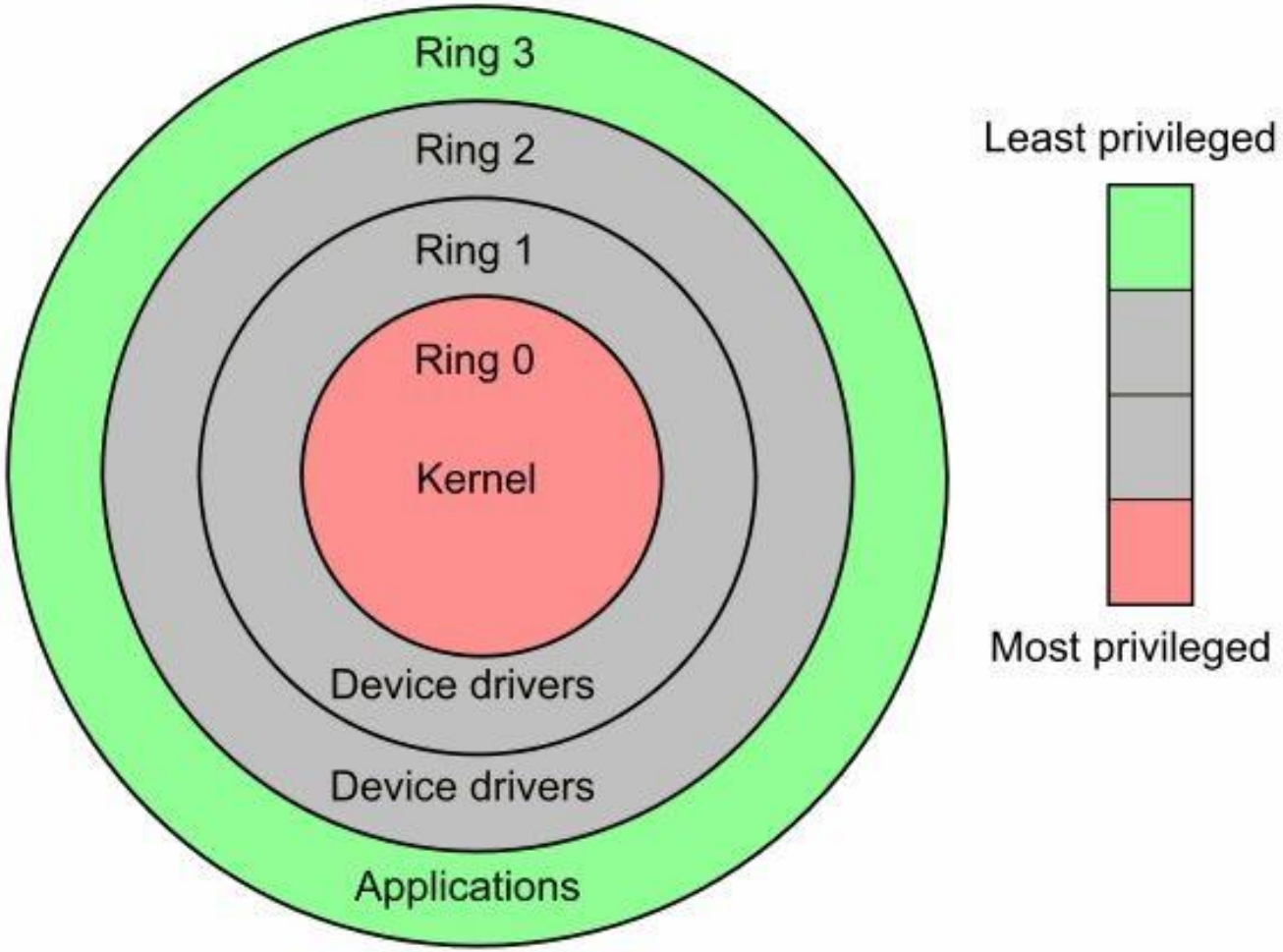
1. **Ring 0 (Kernel Mode)**:
   - Highest privilege level.
   - Direct access to hardware and system resources.
   - Used by the operating system kernel.
2. **Ring 1 and Ring 2**:
   - Intermediate privilege levels.
   - Typically used for device drivers or system services that require more privileges than user applications but less than the kernel.
1. **Ring 3 (User Mode)**:
   - Lowest privilege level.
   - User applications run here with restricted access to hardware and system resources.

Access Example in Ring 0

- **System Calls**: When a user application (Ring 3) needs to perform a privileged operation (like accessing hardware), it makes a system call. This transitions control to the kernel (Ring 0).

CPU Virtualization Techniques & Protection Ring

The x86 architecture offers **four levels of privilege** known as **Ring 0, 1, 2 and 3** to operating systems and applications to **manage access to the computer hardware**



CPU Virtualization Techniques & Protection Ring

User-Space Drivers (Ring 2)

1.USB Drivers:

1. Some USB device drivers operate in user space, allowing for easier updates and less risk to system stability. For example, drivers for certain USB audio devices can run in user mode.

2.Graphics Drivers:

1. Certain graphics drivers, like those using the **Mesa 3D** library, can run in user space, providing flexibility and easier debugging.

3.Network Drivers:

1. Some network drivers, such as those using **User Datagram Protocol (UDP)** for specific applications, can be implemented in user space.

CPU Virtualization Techniques & Protection Ring

Kernel-Space Drivers (Ring1)

1.Block Device Drivers:

1. Drivers for hard drives and SSDs typically run in kernel space to manage direct access to the hardware and ensure high performance.

2.Network Interface Card (NIC) Drivers:

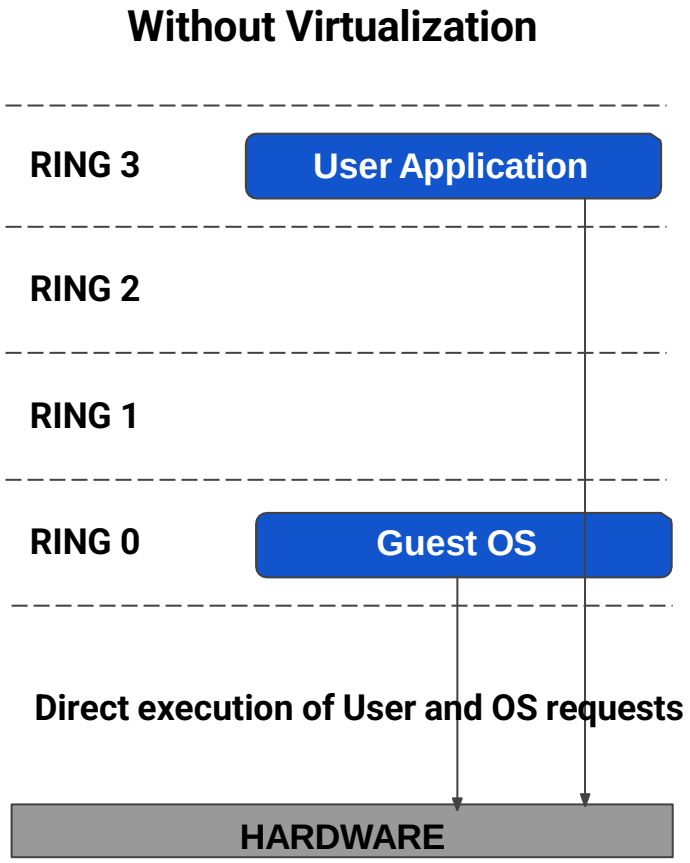
1. Most NIC drivers operate in kernel space to handle packet transmission and reception efficiently.

3.File System Drivers:

1. Drivers that manage file systems (like ext4 or NTFS) run in kernel space to interact directly with the storage hardware.

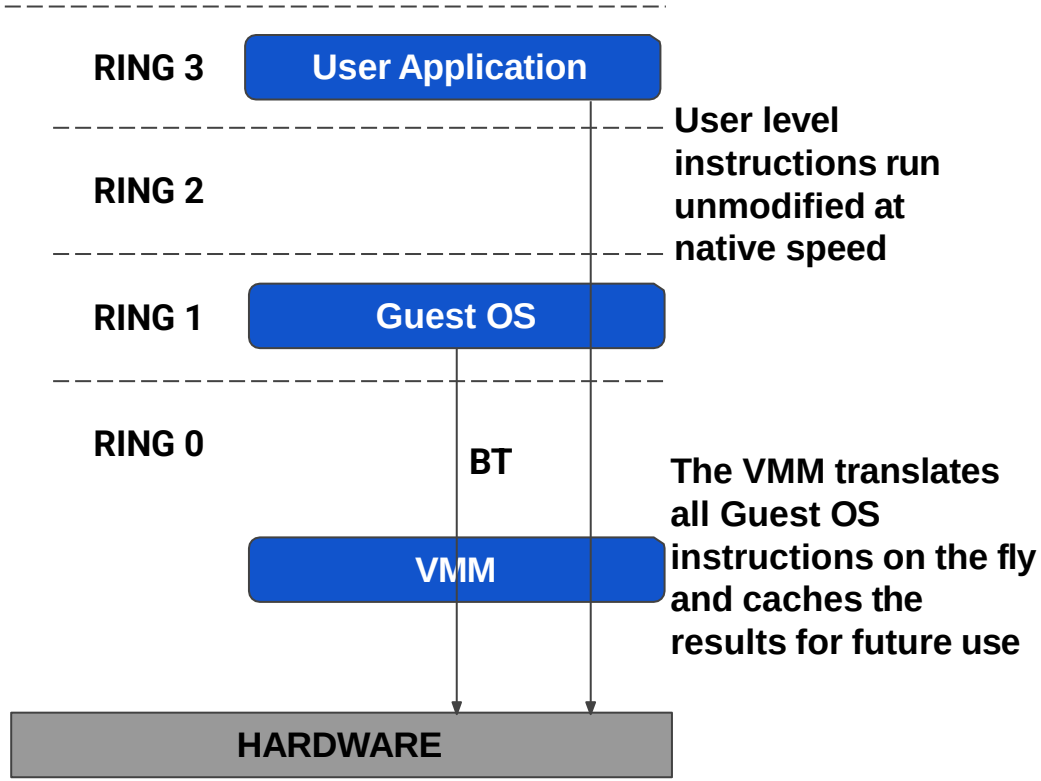
CPU Virtualization Techniques & Protection Ring

The x86 architecture offers **four levels of privilege** known as **Ring 0, 1, 2 and 3** to operating systems and applications to **manage access to the computer hardware**

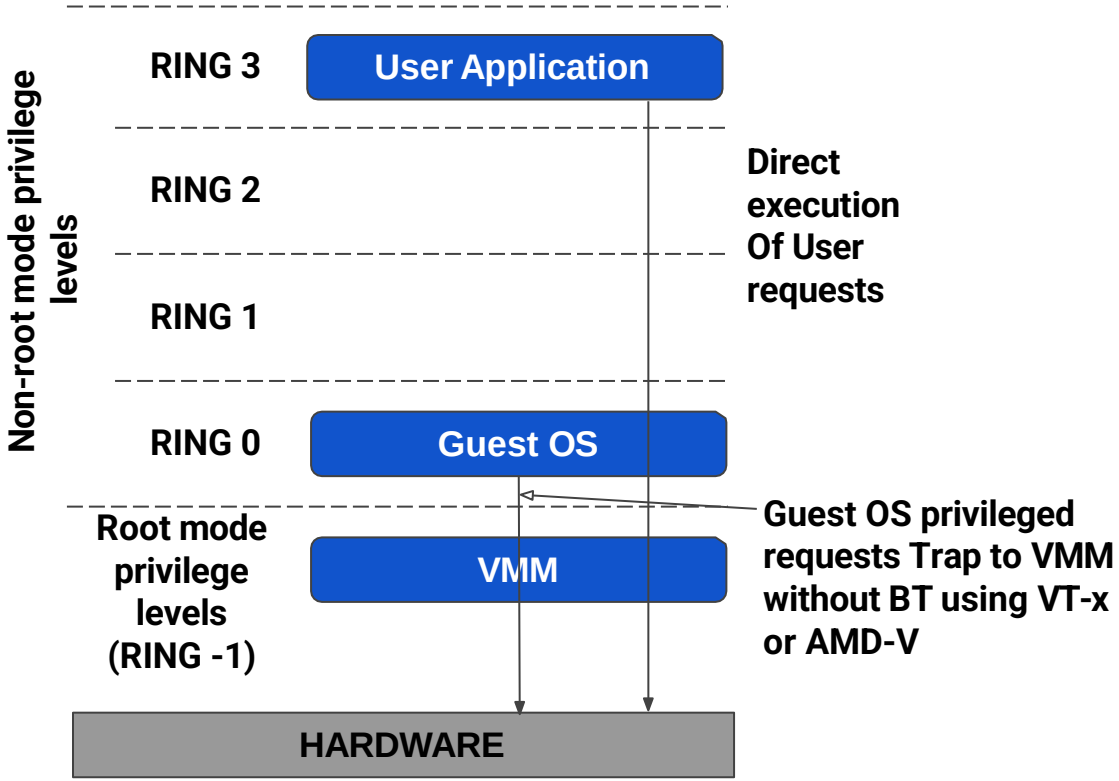


CPU Virtualization Techniques & Protection Ring

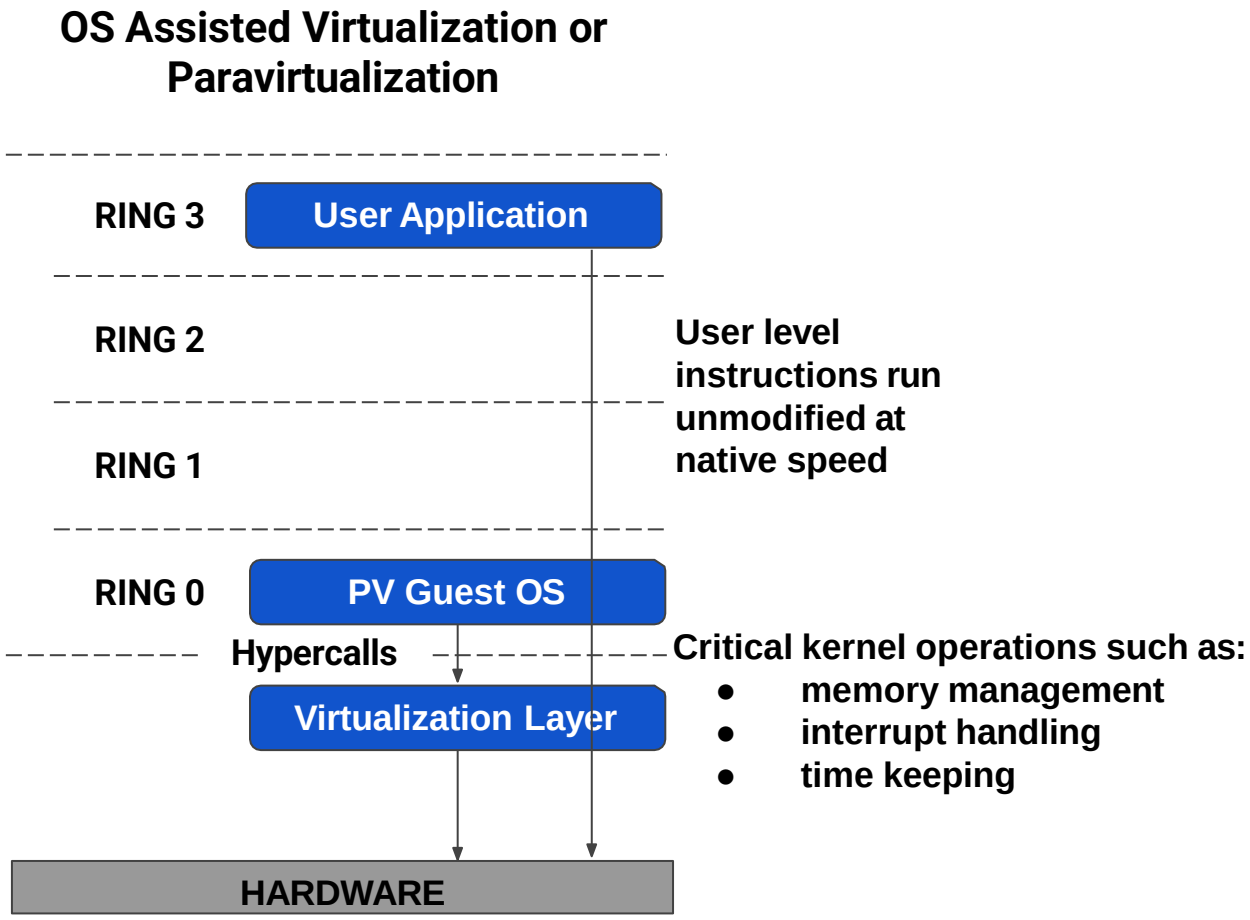
Full Virtualization - Binary Translation (BT)



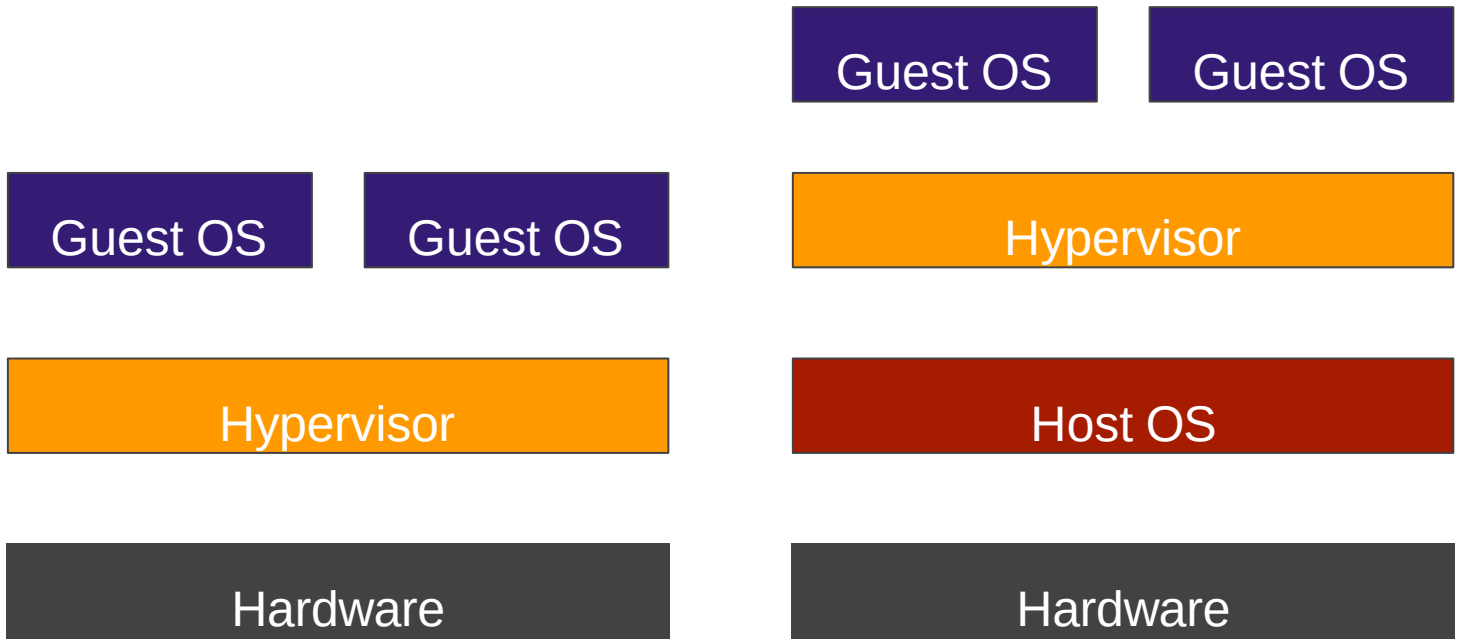
Full Virtualization - Hardware Assisted (VT)



CPU Virtualization Techniques & Protection Ring



How it works ?



TYPE 1 HYPERVISOR

Baremetal

TYPE 2 HYPERVISOR

Hosted

Why call it a “Hypervisor”?

Initially, it was about resource allocation, and the goal was to try to utilize areas of memory that were not normally accessible to programmers. The code produced was successful and was dubbed a hypervisor because, at the time, operating systems were called supervisors, and this code could supersede them.

Bare metal vs hosted hypervisor

- Bare metal or Type 1 hypervisor runs directly on the host's hardware without an underlying operating system.
- Type 1 generally offers better performance and efficiency since it has direct access to hardware resources.
- Type 1 is ideal for data centers, enterprise environments, and scenarios requiring high performance and scalability.
- Hosted hypervisor or Type 2 hypervisor runs on top of an existing operating system.
- Type 2 has slightly lower performance due to the overhead of the host operating system.
- Suitable for desktop virtualization and development environments.

VMM examples

Examples of **Type 1 VMM/hypervisors**:

- Xen (open source)
- Citrix/XenServer
- VMware ESXi
- Microsoft Hyper-V

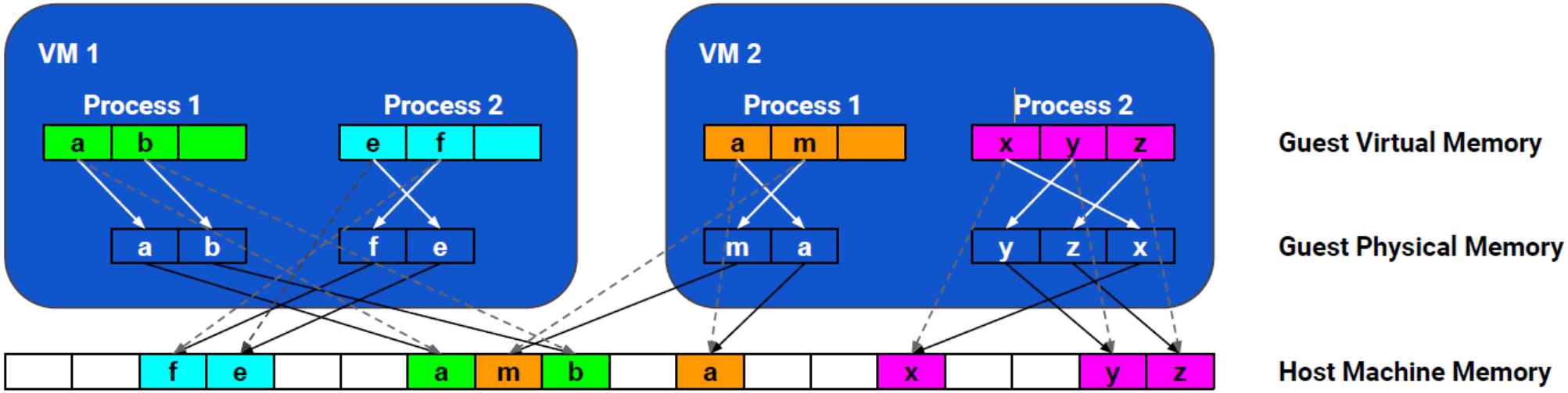
Examples of **Type 2 VMM/hypervisors**:

- QEMU/KVM (open source)
- Oracle VirtualBox (open source)
- Firecracker/KVM (open source)
- Intel NEMU/QEMU/KVM (open source)
- VMware Workstation
- Oracle VM Server for x86
- Oracle Solaris Zones
- Parallels Desktop for MAC

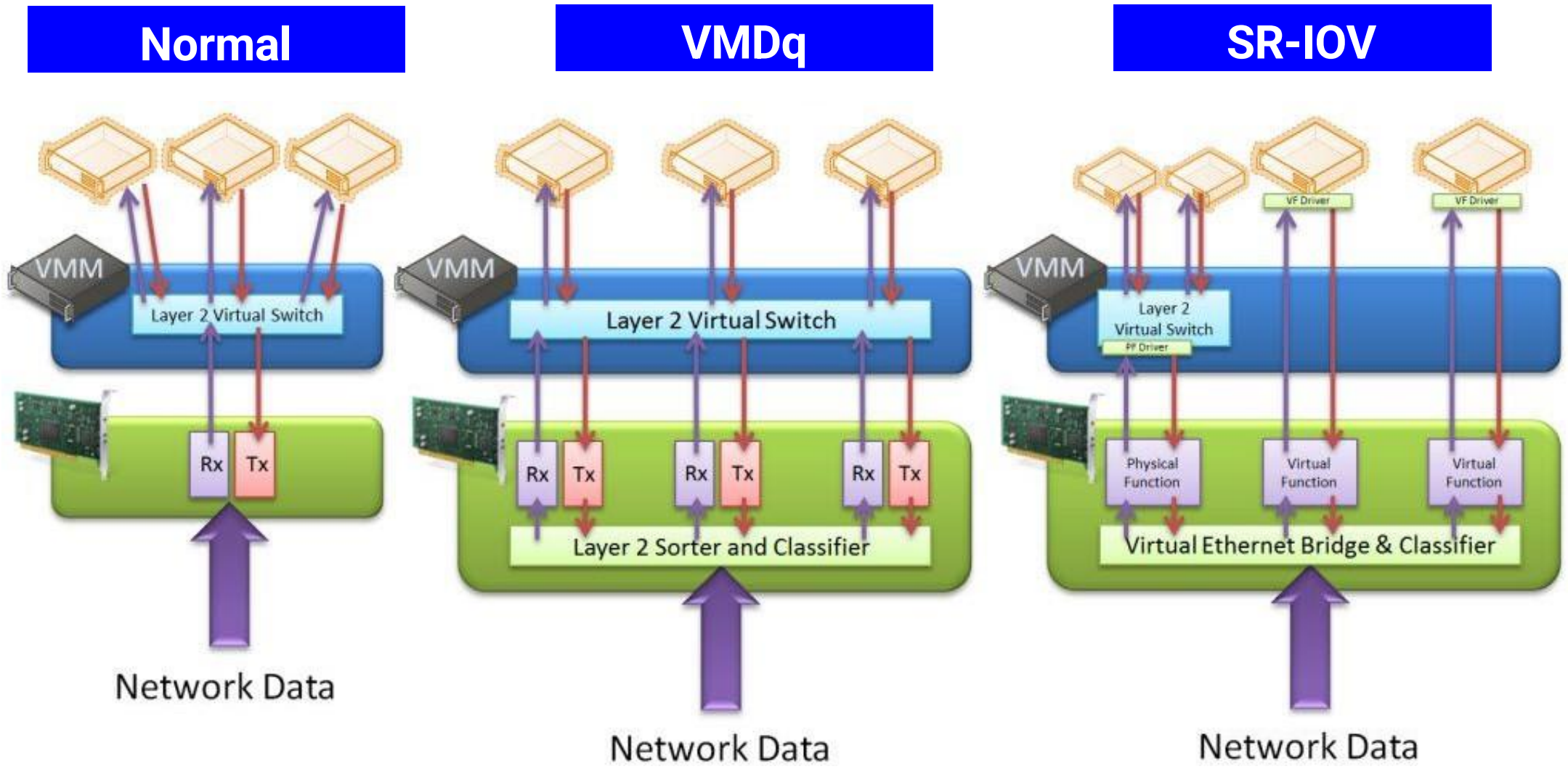
Type 1 & 2 VMMs:

- Linux KVM (Kernel-based Virtual Machine)

Memory Virtualization



Network Interface Virtualization

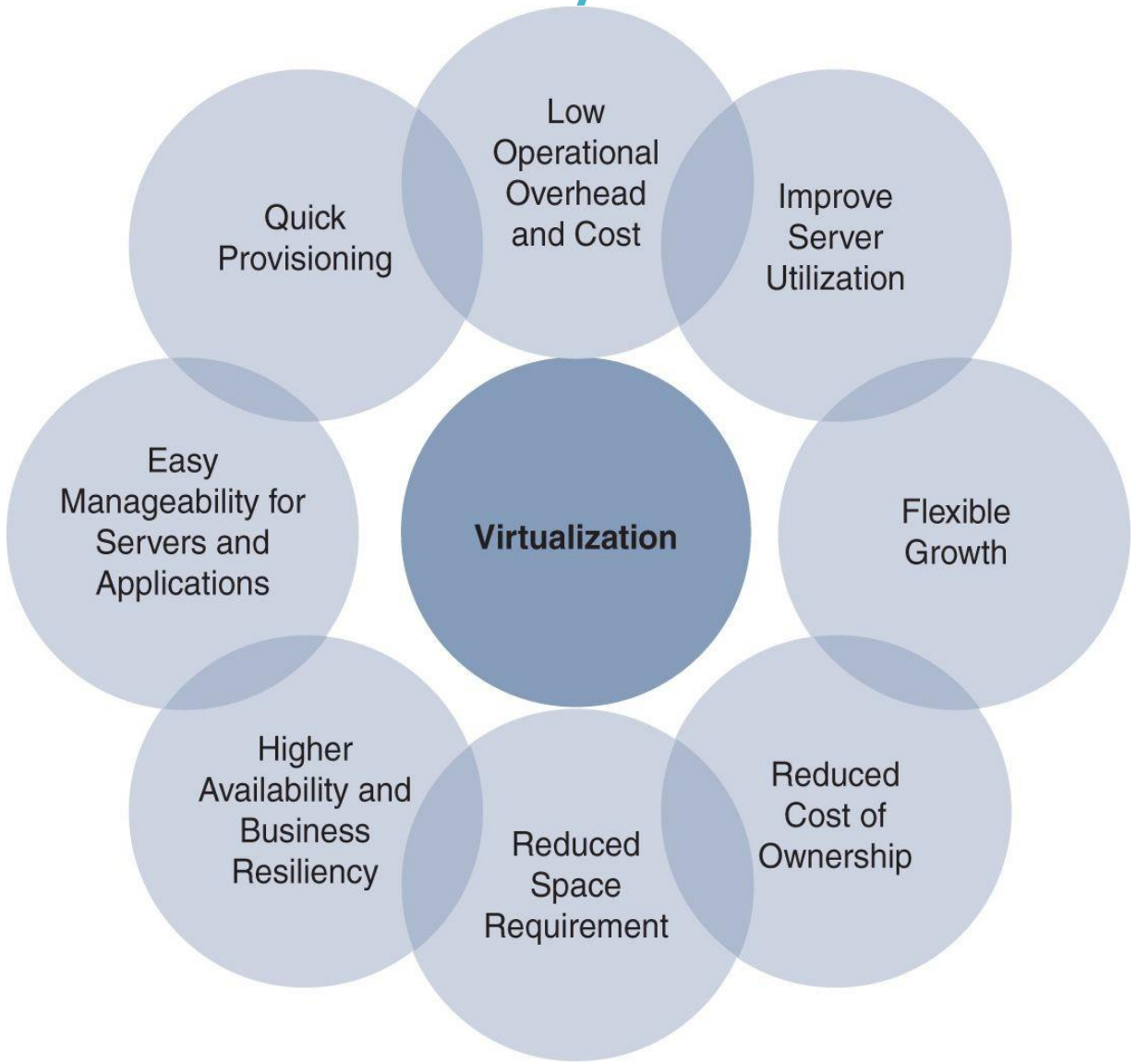


Network Interface Virtualization

1. **Normal Virtualization**
   - The hypervisor manages network traffic and allocates bandwidth to each VM. Each VM communicates through virtual network interfaces (vNICs).
2. **VMDq (Virtual Machine Device Queues)**
   - VMDq is a technology that allows a NIC to maintain multiple receive queues for different VMs, improving the efficiency of network traffic handling.
3. **SR-IOV (Single Root I/O Virtualization)**
   - SR-IOV is a more advanced technology that allows a single physical NIC to present itself as multiple virtual NICs directly to VMs

Feature	Normal Virtualization	VMDq	SR-IOV
NIC sharing	Softwre-based	Multiple receive queues	Virtual functions
Performance	Moderate	Improved	Near-native
CPU overhead	Higher	Reduced	minimal
Complexity	Low	Moderate	High

Virtualization/Softwarization benefits

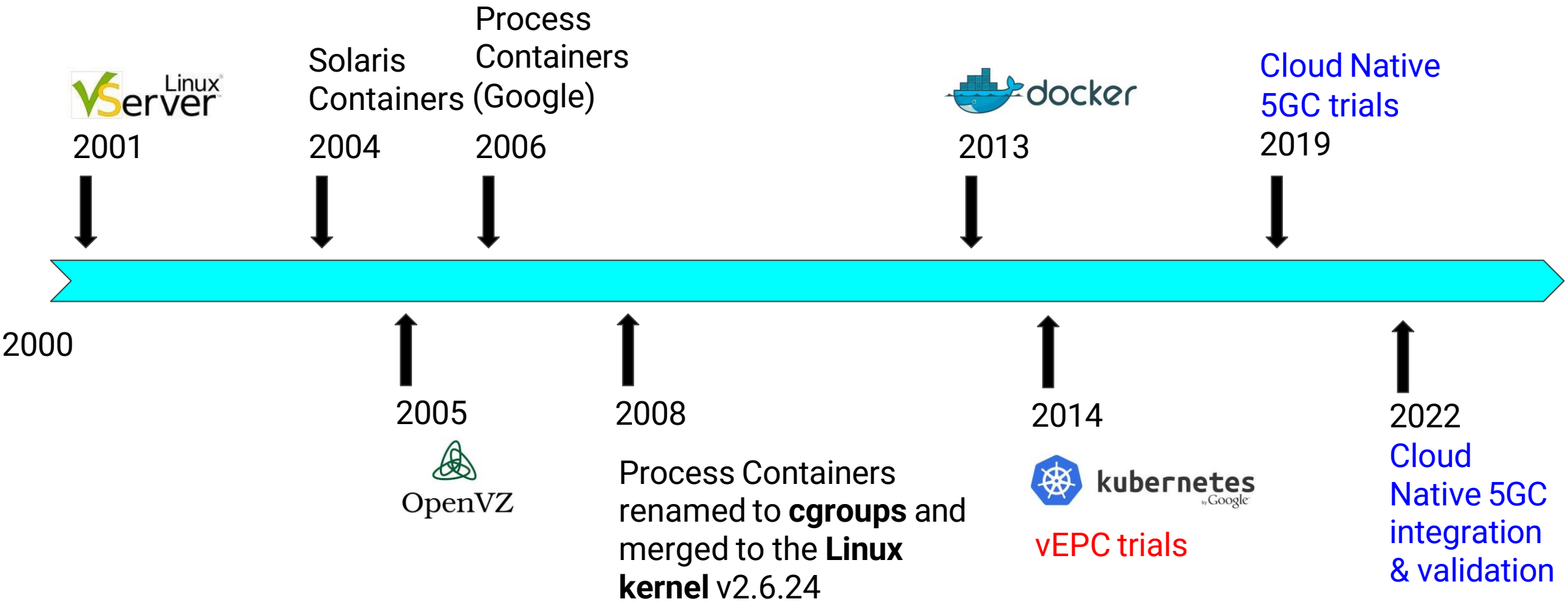


Virtualization can be applied to:

- **Servers**
- **Desktops** (e.g. Citrix XenDesktop, VMware Horizon View)
- **Software**
- **Networks**

https://create.kahoot.it/preview/7b9810d5-6f23-4c58-9797-1c61cd232505

OS Level Virtualization: a bit of history



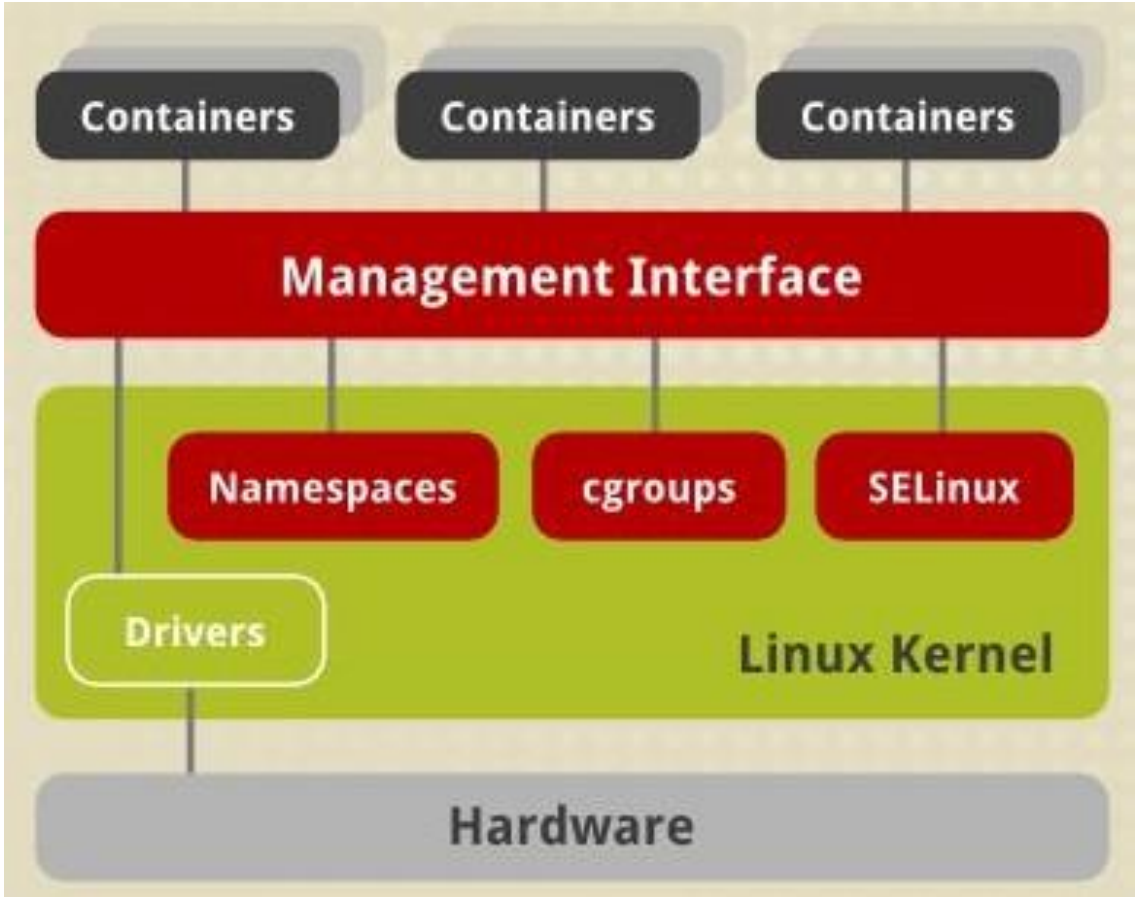
What are containers? [11]

- Containers offer a **logical packaging mechanism** in which **applications can be abstracted from the environment in which they actually run**. This decoupling allows container-based applications to be deployed easily and consistently, regardless of whether the target environment is a private data center, the public cloud, or even a developer's personal laptop

- Containerization **provides a clean separation of concerns**, as **developers focus on their application logic and dependencies**, while **IT operations teams can focus on deployment and management** without bothering with application details such as specific software versions and configurations specific to the app

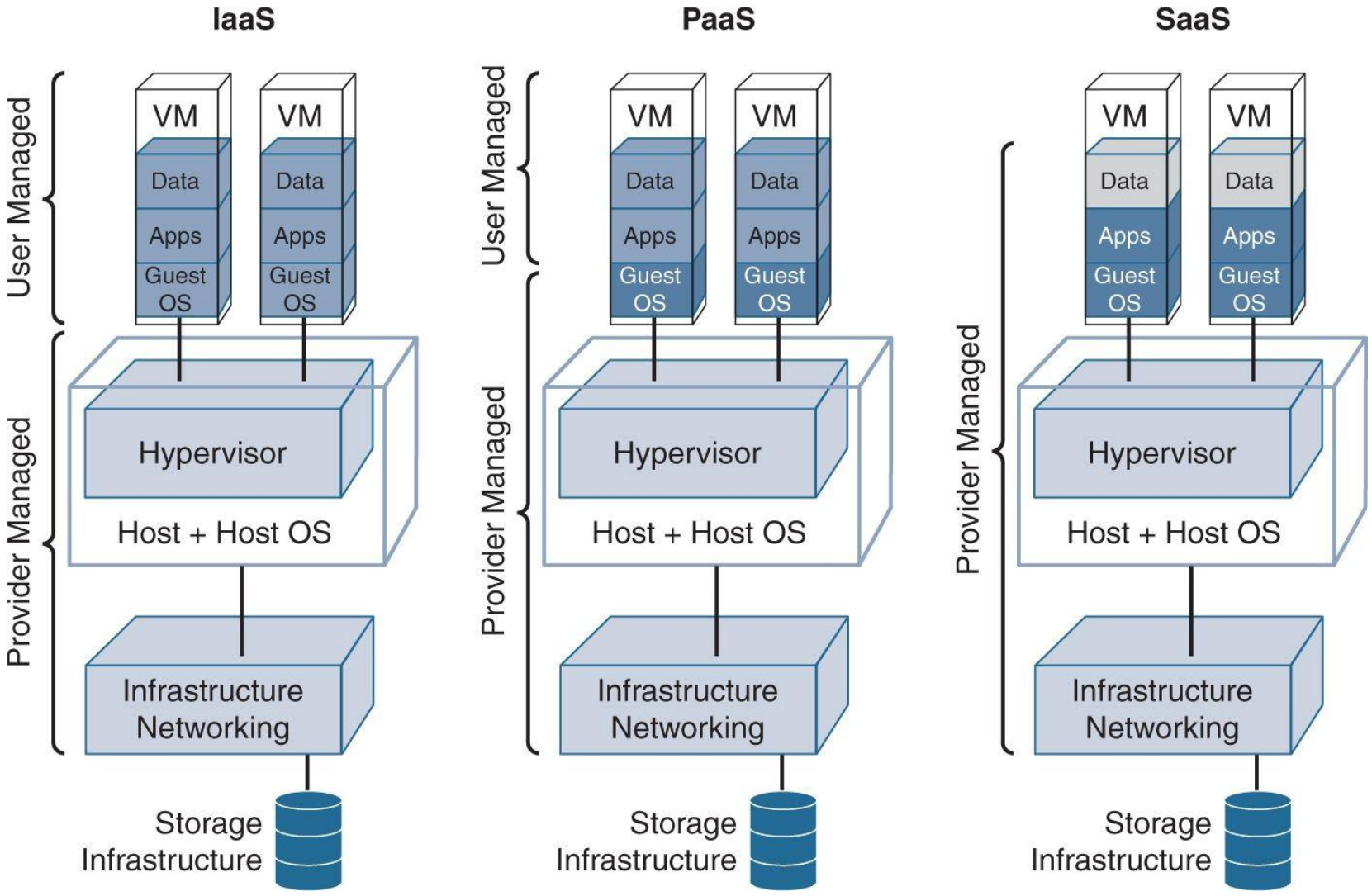
Linux Cgroups, Namespaces, SELinux

- **Namespaces**: abstract a particular global system resource and make it appear as a separated instance to processes within a namespace (e.g. Network, Mount, PID, IPC namespace, etc.)
- **Cgroups**: allows limiting and monitoring CPU time, system memory, network bandwidth of a set of processes
- **SELinux** (Security-Enhanced Linux): enforces container isolation by applying SELinux policy and labels

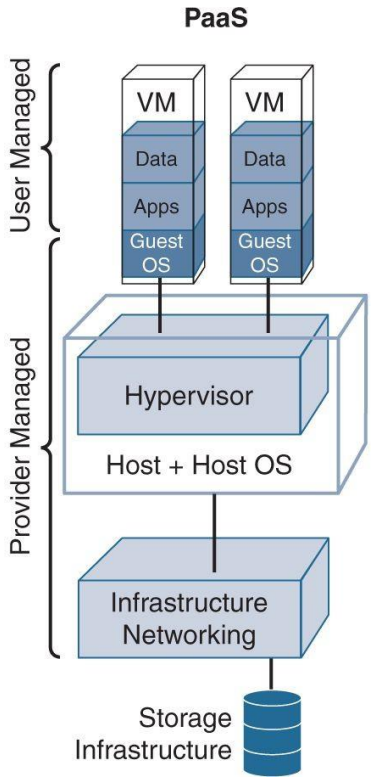


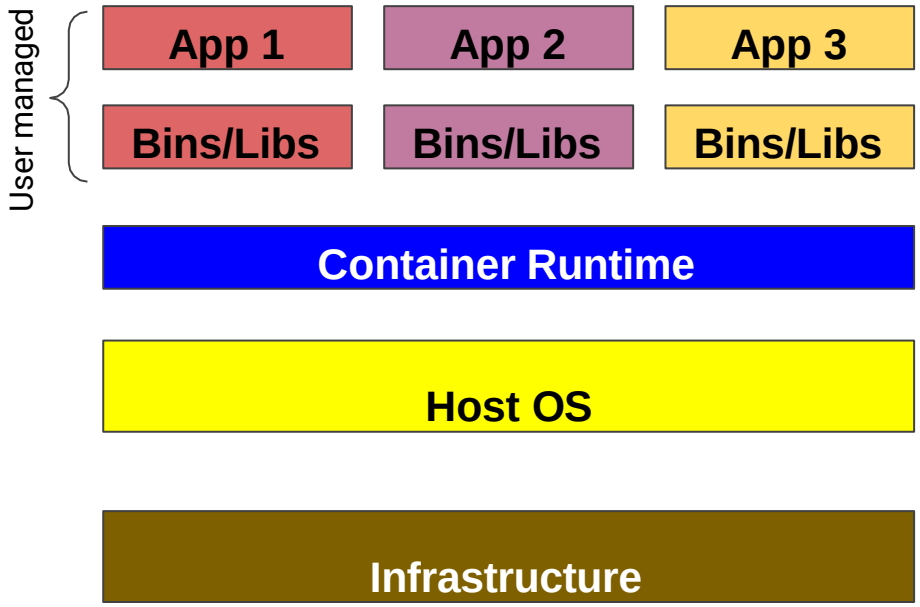
redhat.com

Cloud Computing Models: IaaS, PaaS, SaaS

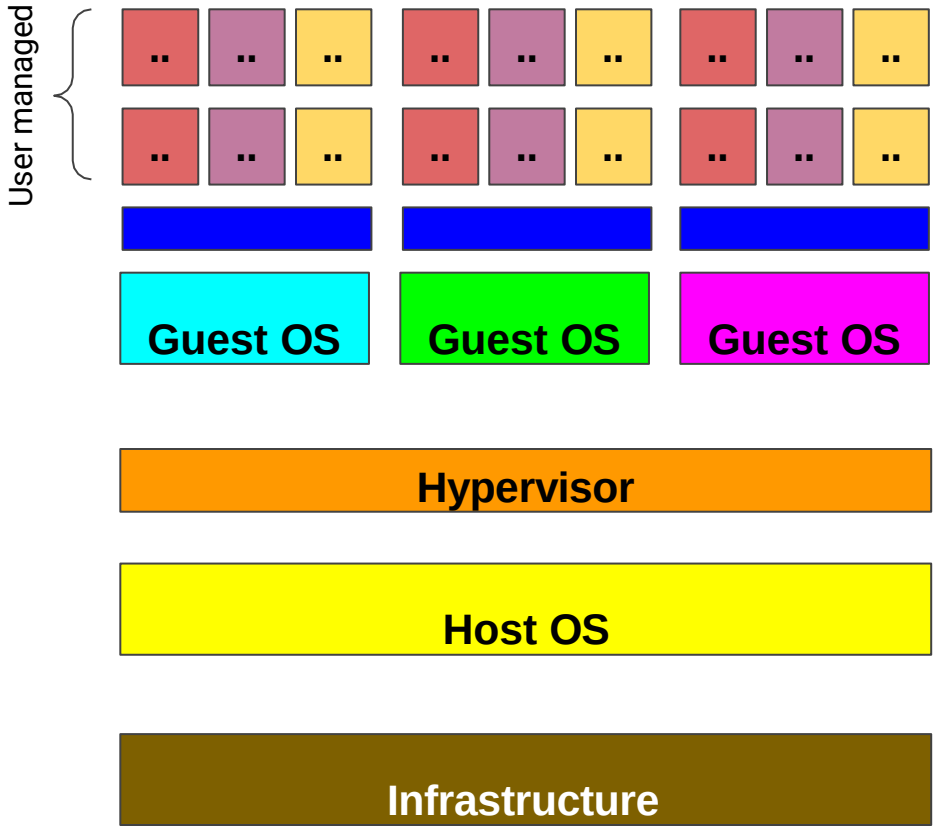


Container-as-a-Service (CaaS)





Option 1: Container on Baremetal



Option 2: Container in VM

Exercise: container vs VM ?

Containerization Benefits

- Greater density
  - several GB in size for a VM vs few dozen of MB for a Container
  - a maximum of 6000 containers can run theoretically on a host machine
- Faster to launch and easy to scale up/down
  - a container is just another process running on the Host OS
- Uses far fewer resources than a VM
  - higher utilization of compute resources than traditional or hardware VMs
- Well adapted for microservice architecture and design pattern
- Lower I/O latency and CPU overhead
- Increased portability
  - application container can run in a predictable/reproducible way on different OSs and environments
- Operational simplicity
  - Container engines provide a very simple yet powerful CLIs to Life Cycle Manage containers (create, start, stop, scale, destroy, attach, etc.)