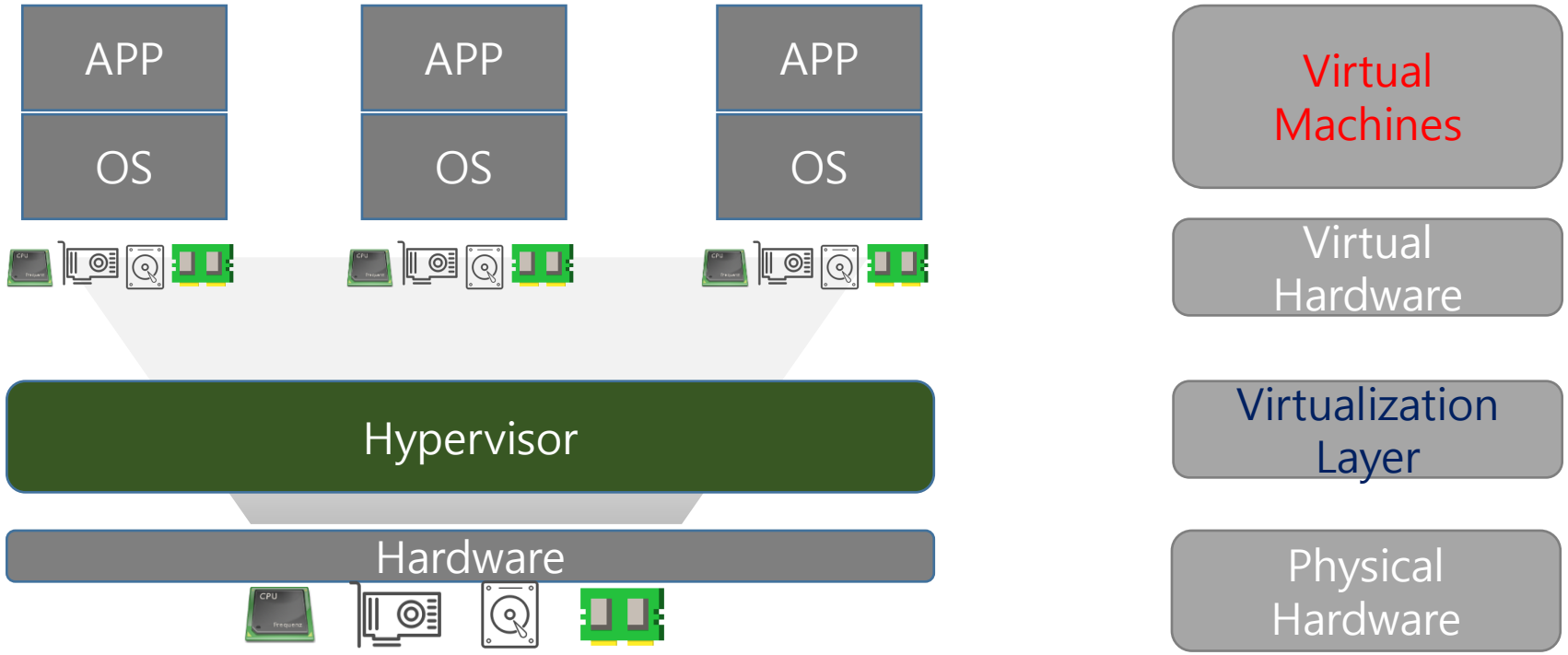
Virtualization 1.
(가상화)
- HW virtualization for VM(Virtual Machine)

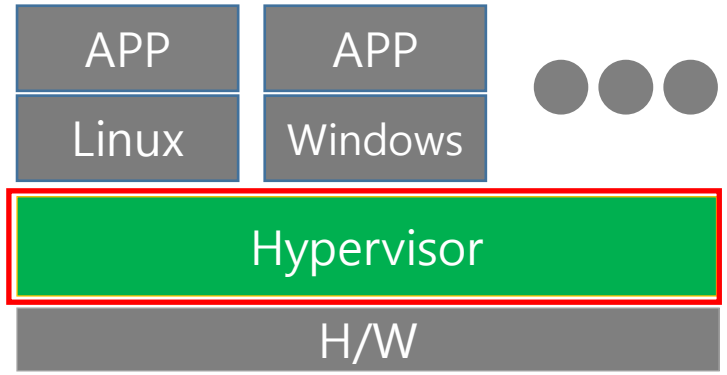
KVM, QEMU, Libvirt 가 클라우드관련 오픈소스라서 많이 사용되는데..
어떤 역할을 하지?

- Virtualization
  - 컴퓨터에서 CPU, Memory, I/O등의 물리적인 리소스를 추상화(Abstraction)하는, 즉 물리적인 자원을 논리적인 자원으로 보이게 하는 기술

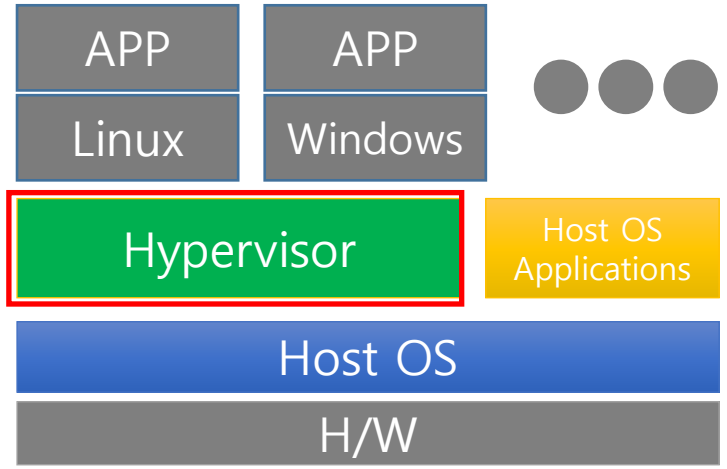
- Hypervisor
  - 호스트 컴퓨터에서 다수의 운영 체제(operating system)를 동시에 실행하기 위해 하드웨어 자원을 논리적 플랫폼(platform)으로 가상화해 주는 소프트웨어
  - 논리적인 자원으로 가상화된 환경 위에 여러 OS를 동시에 운영할 수 있다.



Hypervisors 분류 : 설치방식에 따른 분류



- Type 1 (Native or Bare metal)
  - 호스트의 하드웨어 상에서 바로 동작하며, 게스트 OS를 관리
  - CPU가 가상화를 지원해야 함
  (Intel VT, AMD-V 기능 등)
  - XenServer or Vmware ESX, Hyper-V(MS사)
  KVM(Linux에 같이 포함됨)

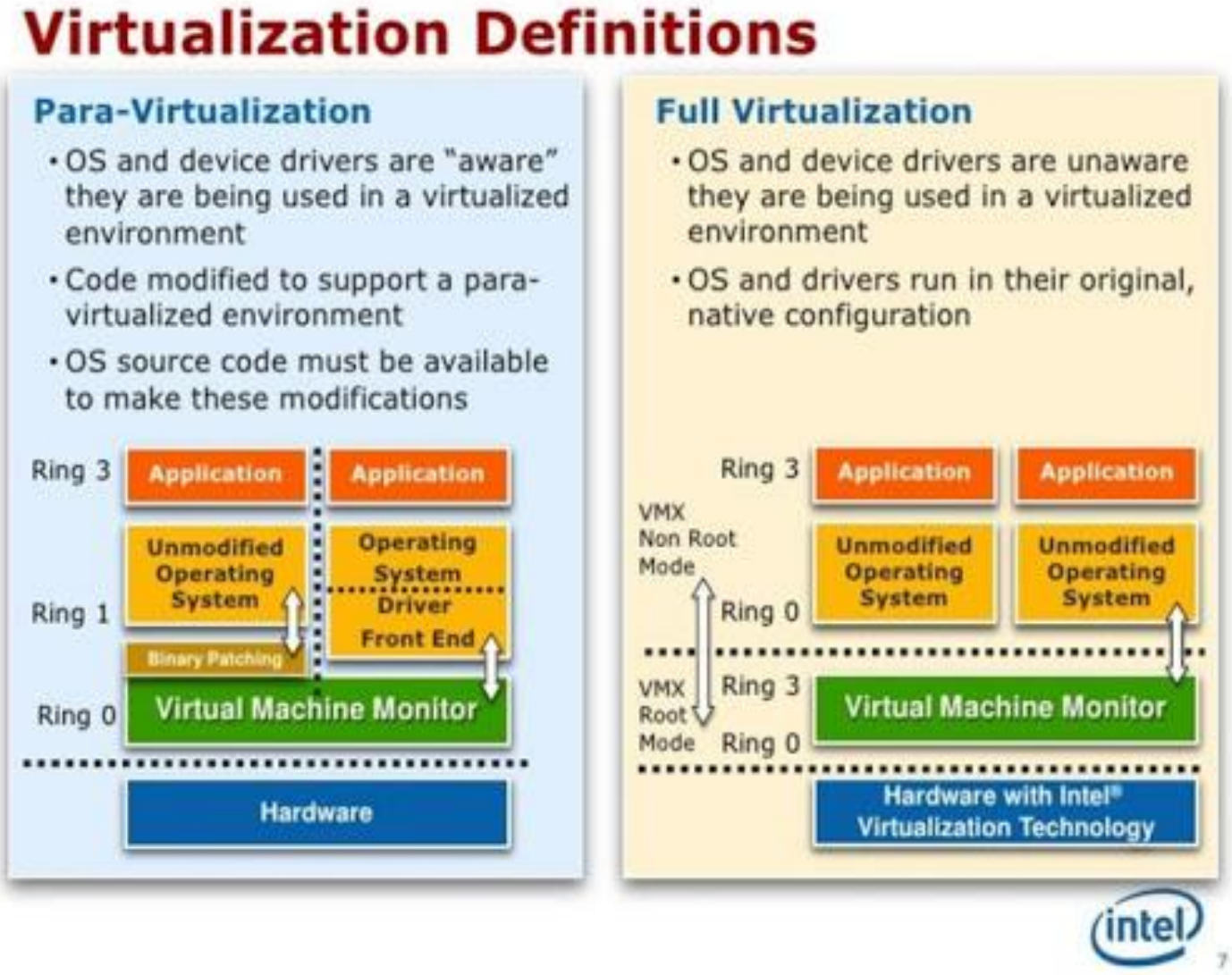


- Type 2 (Hosted)
  - 호스트의 OS상에서 Hypervisor동작
  - Hypervisor는 OS를 통해 H/W를 호출
  - **VirtualBox**, VMware Desktop
  - **QEMU**

KVM(Kernel based Virtual Machine) 주) www.linux-kvm.org 에서 발췌

KVM is a **Linux kernel module**.
It is a **type 1 hypervisor** that is a **full virtualization solution for Linux on x86 hardware** containing virtualization extensions (Intel VT or AMD-V).
But what is full virtualization, you may ask? **When a CPU is emulated (vCPU) by the hypervisor, the hypervisor has to translate the instructions meant for the vCPU to the physical CPU**. As you can imagine this has a massive performance impact. To overcome this, modern processors support virtualization extensions, **such as Intel VT-x and AMD-V**. **These technologies provide the ability for a slice of the physical CPU to be directly mapped to the vCPU**. Therefor the instructions meant for the vCPU can be directly executed on the physical CPU slice.

Full Virtualization vs Para Virtualization



Hypervisor 분류 2: 어떤 방식으로 가상화를 했는가?
(전가상화(Full Virtualization), 반가상화(Para Virtualization)

- **Full Virtualization(전가상화)**

  하드웨어 전체를 완전히 가상화- 그래서 Guest OS를 원 OS의 변경없이 사용가능

- **Para Virtualization(반가상화)**

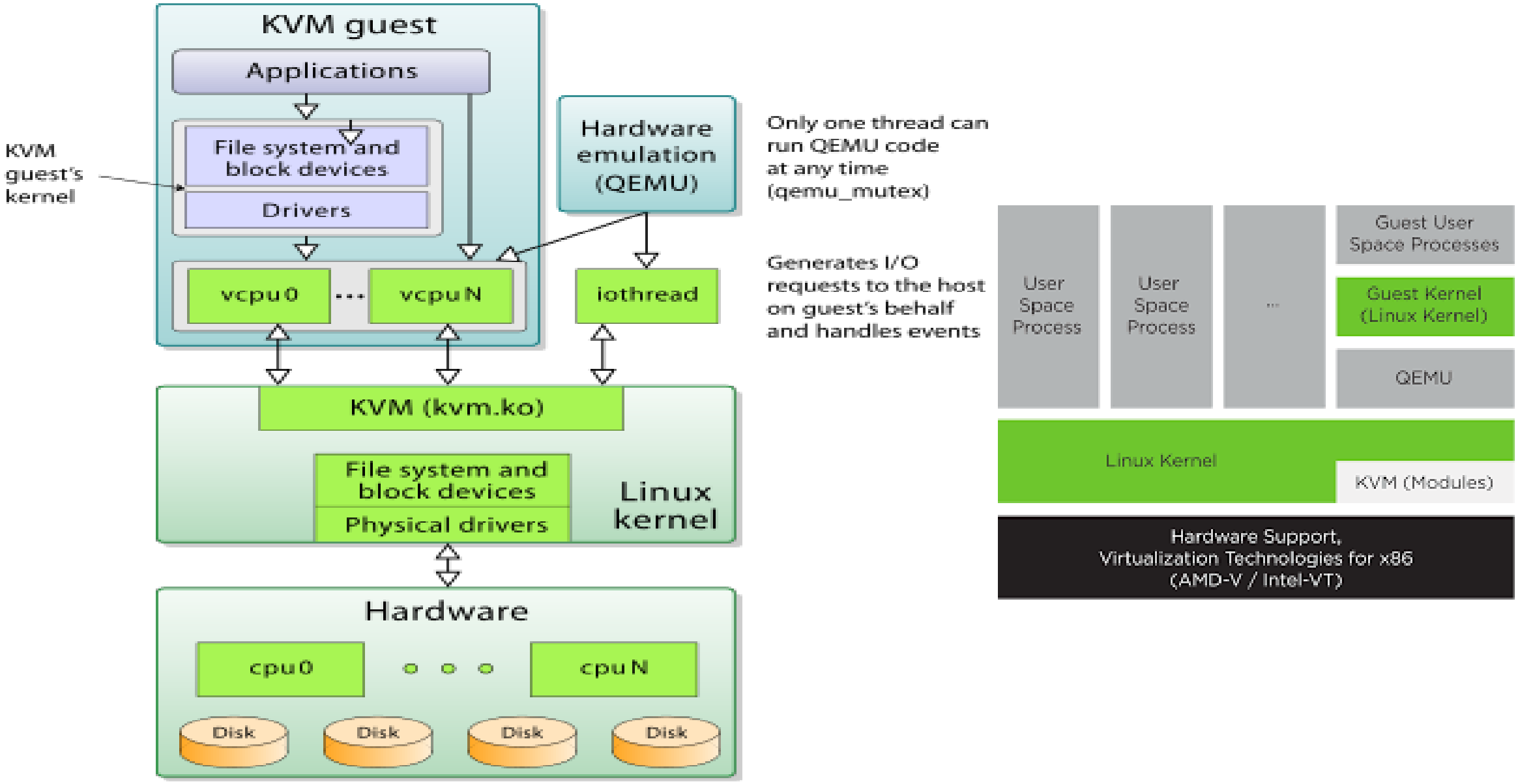
  하드웨어를 완전히 가상화하지 않음- Guest OS에서 직접 하드웨어 제어가 안되고 하이퍼바이저를 통해 제어

  그래서 **Guest OS는 원래 OS로부터 일부 수정된 것들이 사용됨**

  예) QEMU, QEMU와 함께 KVM을 운영할 수 있음..

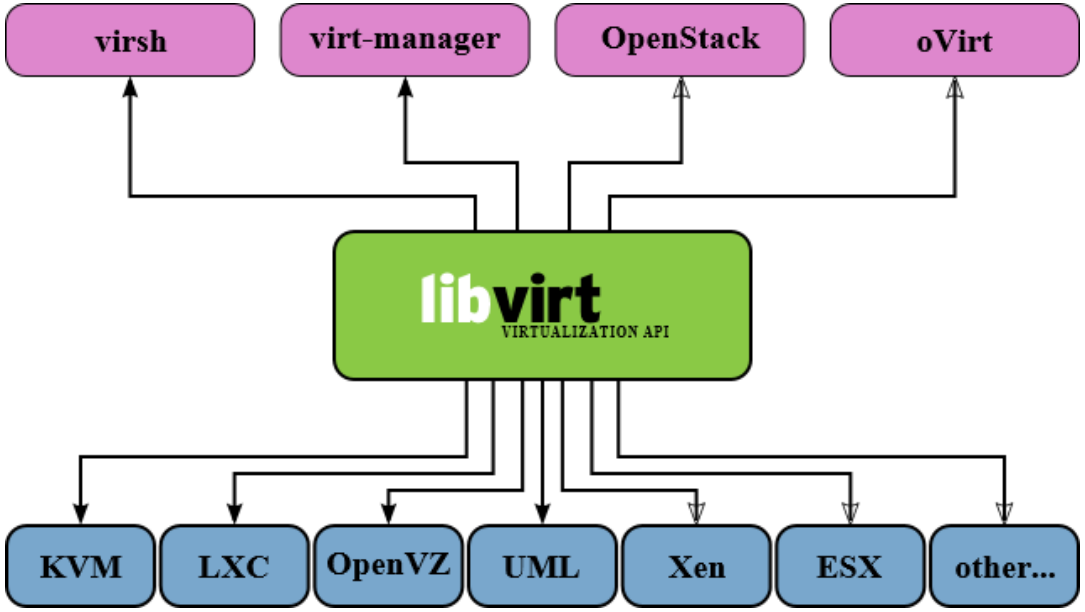
QEMU(Quick Emulator)-
HW emulator

- QEMU is a **type 2 (i.e runs upon a host OS) hypervisor** for performing hardware virtualization (**not** to be confused with **hardware-assisted virtualization**), such as disk, network, VGA, PCI, USB, serial/parallel ports, etc. It is flexible in that it can emulate CPUs via dynamic binary translation (DBT) allowing code written for a given processor to be executed on another (i.e ARM on x86, or PPC on ARM). Though QEMU can run on its own and emulate all of the virtual machines resources, as **all the emulation is performed in software it is extremely slow**.
- QEMU can run independently, but due to the emulation being performed entirely in software it is extremely slow. To overcome this, **QEMU allows you to use KVM as an accelerator so that the physical CPU virtualization extensions can be used.** So to conclude QEMU is a type 2 hypervisor that runs within user space and performs virtual hardware emulation, where as KVM is a type 1 hypervisor that runs in kernel space, that allows a user space program access to the hardware virtualization features of various processors.


KVM guest
Applications
File system and block devices
Drivers
KVM guest's kernel
vcpu 0
vcpu N
Hardware emulation (QEMU)
iothread
Only one thread can run QEMU code at any time (qemu_mutex)
Generates I/O requests to the host on guest's behalf and handles events
KVM (kvm.ko)
File system and block devices
Physical drivers
Linux kernel
Hardware
cpu 0
cpu N
Disk
Disk
Disk
Disk
User Space Process
User Space Process
...
Guest User Space Processes
Guest Kernel (Linux Kernel)
QEMU
Linux Kernel
KVM (Modules)
Hardware Support, Virtualization Technologies for x86 (AMD-V / Intel-VT)

Libvirt

- **Libvirt** is a **virtualization management library**. And what does it do? It manages both KVM and QEMU. It consists of three utilities namely – an API library, a daemon (libvirtd) and a command line tool -virsh. Libvirt is quite effective and **it can manage a lot of hypervisors altogether**.(가상화 플랫폼(하이퍼바이저)을 관리하는 오픈소스 라이브러리, 다양한 하이퍼바이저 환경을 통합적으로 관리하기 위해 탄생)



참고자료

1. Open source virtualization
2. Choose between 5 hosted hypervisors based on features, use cases

   Hypervisors such as VMware Workstation, VMware Fusion, VMware Horizon 7, Oracle VM VirtualBox and Parallels Desktop provide value to small data centers in different ways.

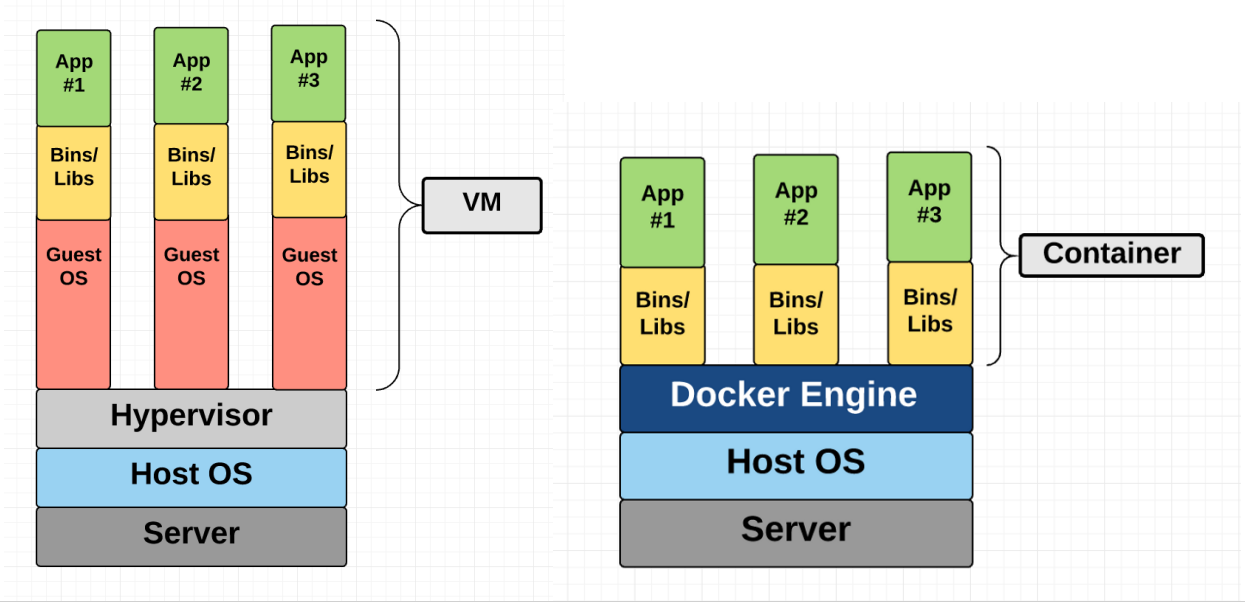
3. Compare the top 5 bare-metal hypervisors

   Hypervisors such as VMware vSphere, Microsoft Hyper-V, Citrix Hypervisor, RHV and Oracle VM Server for x86 provide value to enterprise-level data centers in different ways.

Virtualization 2.
- OS virtualization for Container

VM vs 컨테이너 --- 상세한 내용은 오픈스택 학습후에 진행함

- VM
  - HOST OS 위에서 Hypervisor를 통해 자원을 가상화 하여 VM을 동작
  - HOST OS 위에 Guest OS가 동작하는 구조
- Container
  - HOST OS에서 프로세스를 위한 공간을 별도로 분리
  - 기본적인 Binary, Library 만을 guest os 대신 사용



앞으로 배울 내용 : VM, Container Orchestration(관리) 시스템 – Openstack, Kubernetes

- 수많은 서버가 모여 있는 센터- **IDC (Internet Data Center) / 데이타센터**
  → 수많은 가상머신,컨테이너가 모여 있는 센터 - **클라우드**

- 서버들 속에 hypervisor, Container Engine이 만드는 **VM, Container 운용, 관리 기술**을 공부하는 것이 결국 **클라우드 기술을 학습**하는 것