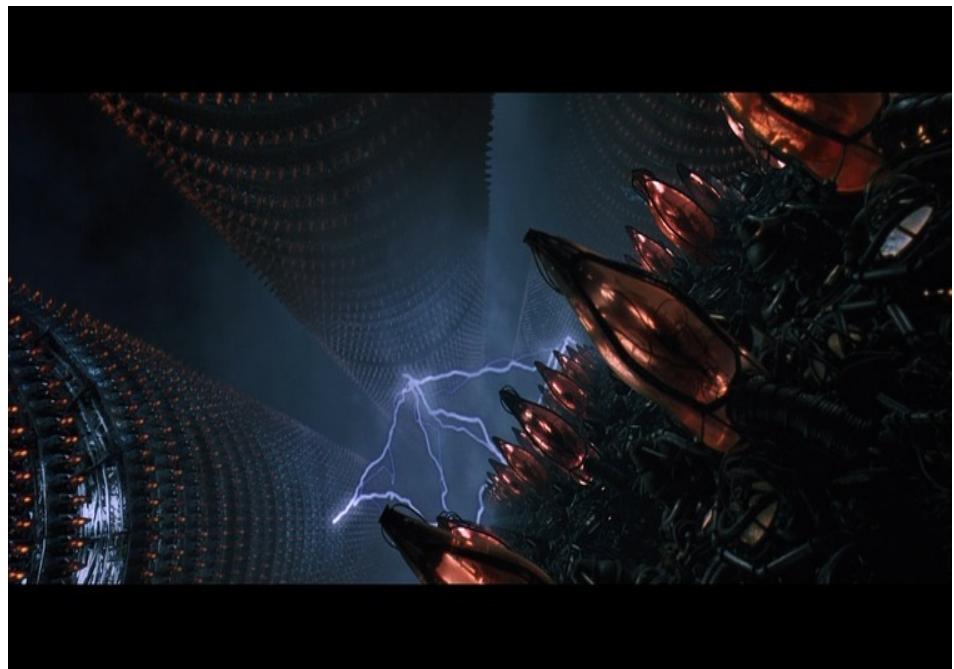




Virtualization

**최신기술 콜로키움
CSE438-00**

Matrix (Movie)



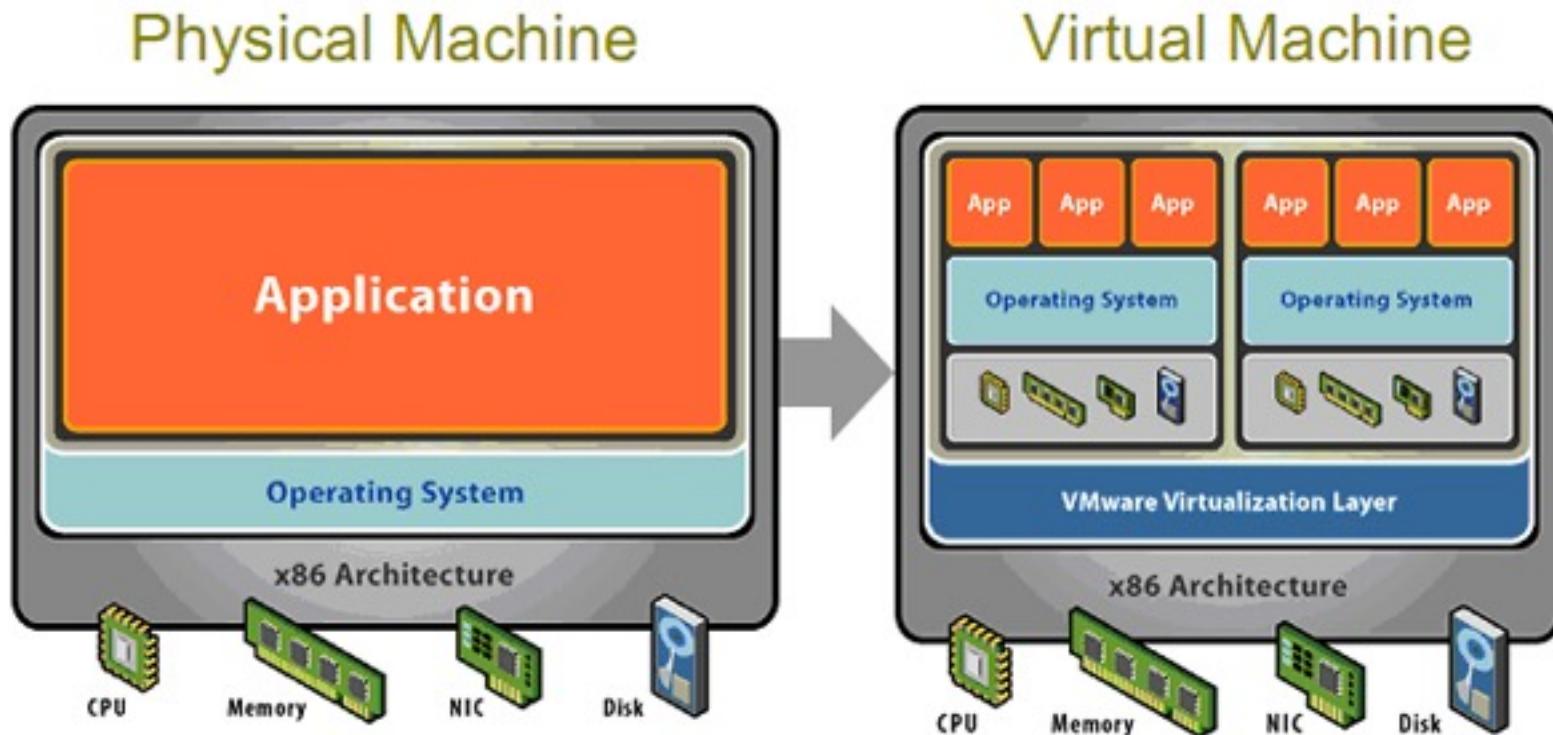
Matrix (Movie)



- Matrix : 기계가 창조한 가상현실
 - Red : Matrix를 벗어난 혹독한 현실로 갈 수 있는 길
 - Blue : 달콤한 가상현실에 남는 길

Virtualization

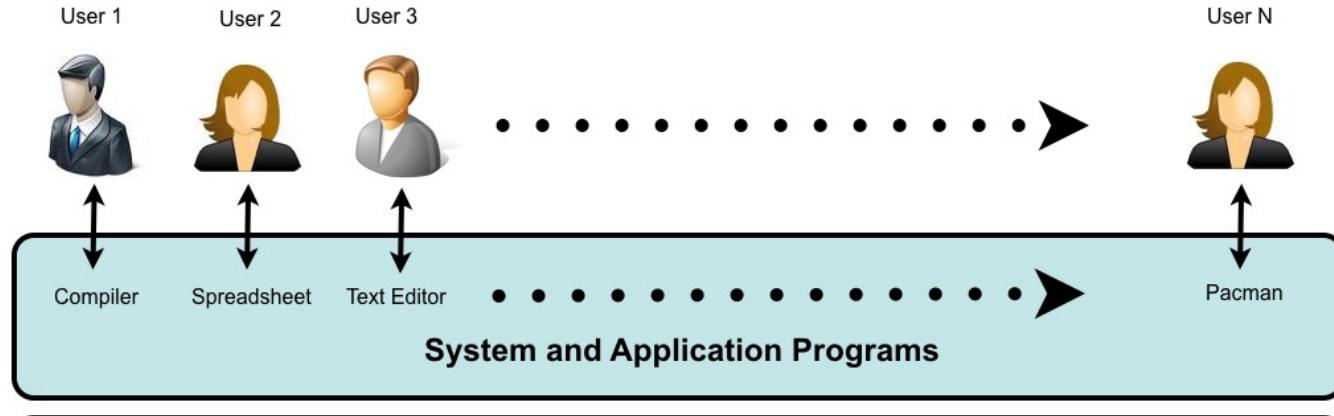
- 가상화
 - 실제 존재하지 않는 것을 존재하는 것처럼 보이게 함
 - 하나의 물리 머신 위에 여러 개의 가상머신을 생성



Why Virtualization?

- **Lower Costs**
 - 모든 서버들이 항상 바쁜 것은 아님 (사용률 10% 이하)
 - 하나의 물리 머신에 여러 대의 가상 머신을 이용 -> 효율성 증가
- **Fast Provisioning**
 - 물리 서버를 설치하는데 많은 시간과 노력이 필요
 - 네트워크 연결, 전원 연결, OS 설치 등등
 - 가상 머신은 데이터 이동만으로 배포/확장 가능
- **Easy Management**
 - 손 쉬운 백업 및 교체 및 모니터링

OS

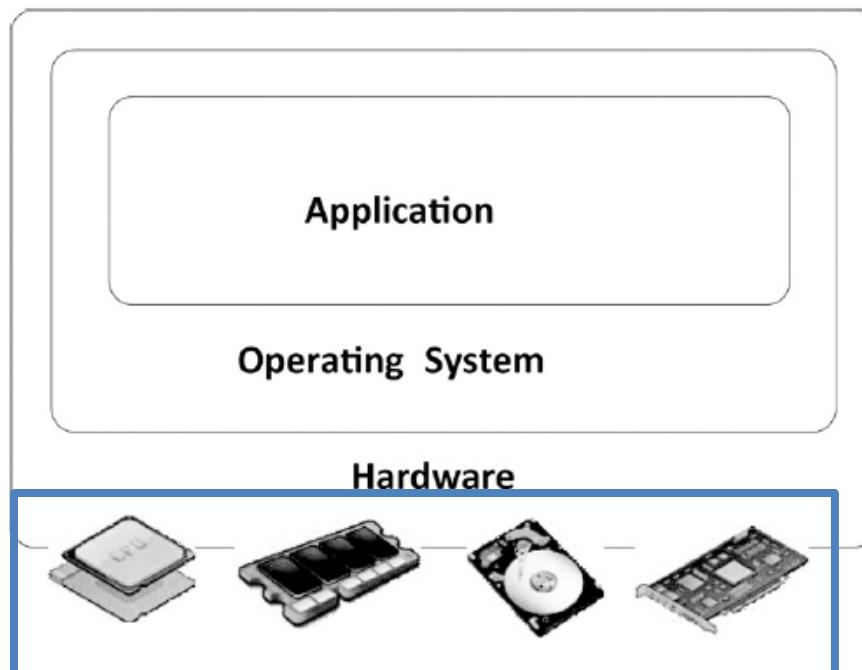


가상머신의 OS에게 진짜 하드웨어 위에서
동작하고 있다고 어떻게 믿게 만들까?

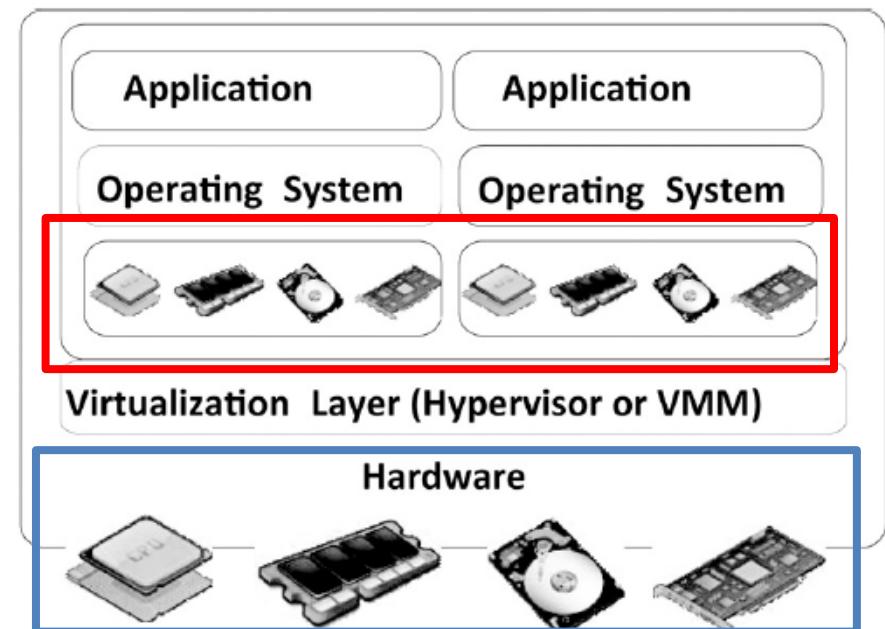


How?

- Hypervisor
 - 가상 머신 OS에게 실제 하드웨어를 가지고 있다고 착각하게 만듦
 - 가상화의 역할을 담당하는 소프트웨어
 - Virtual Machine Monitor (VMM)이라고도 불림



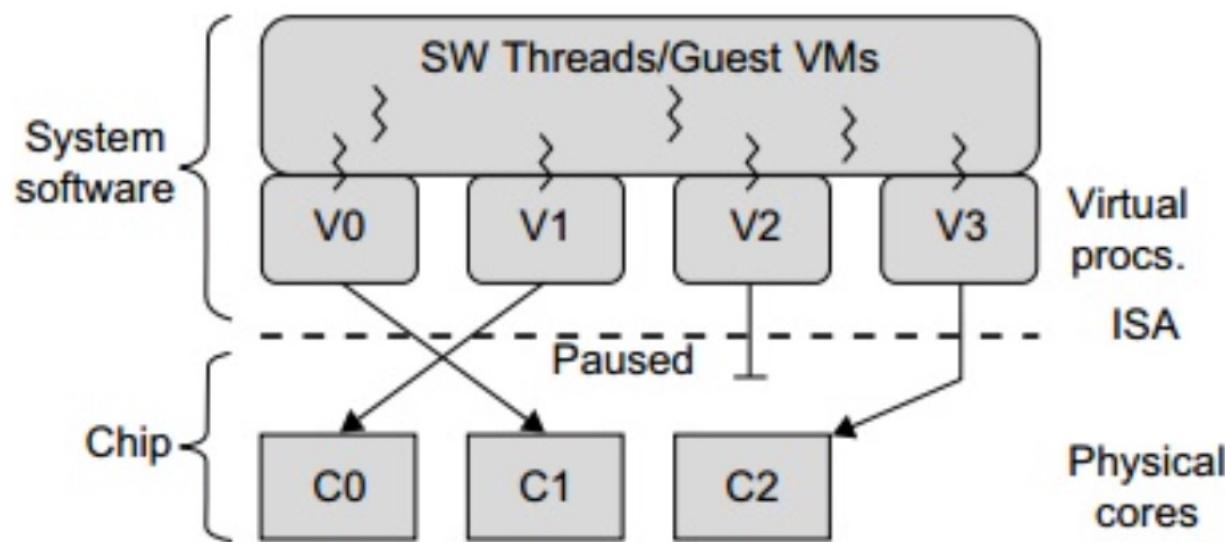
(a) Before Virtualization



(b) After Virtualization

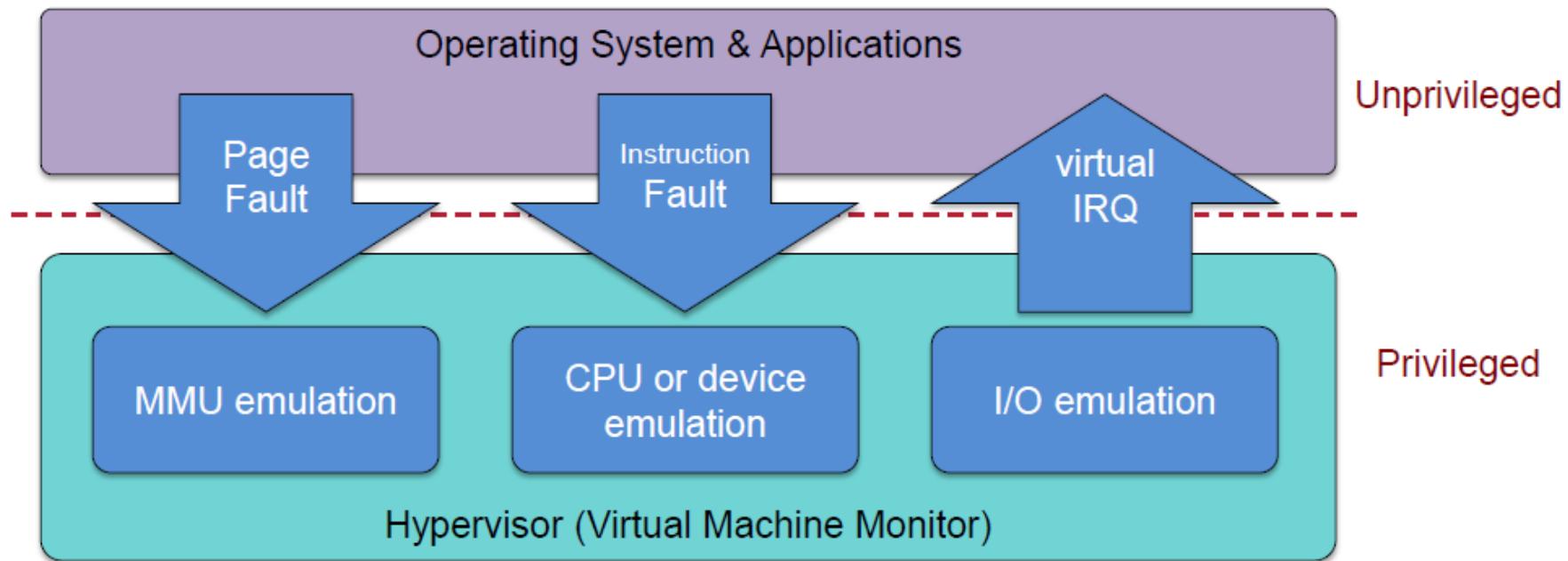
Virtualized CPU

- vCPU
 - Hypervisor가 가상머신의 OS에게 제공하는 CPU
 - Privileged Operation : OS만 수행할 수 있는 명령
 - 가상머신은 Privileged Operation을 수행할 수 있는 권한 없음
 - Hypervisor가 이를 대신 수행



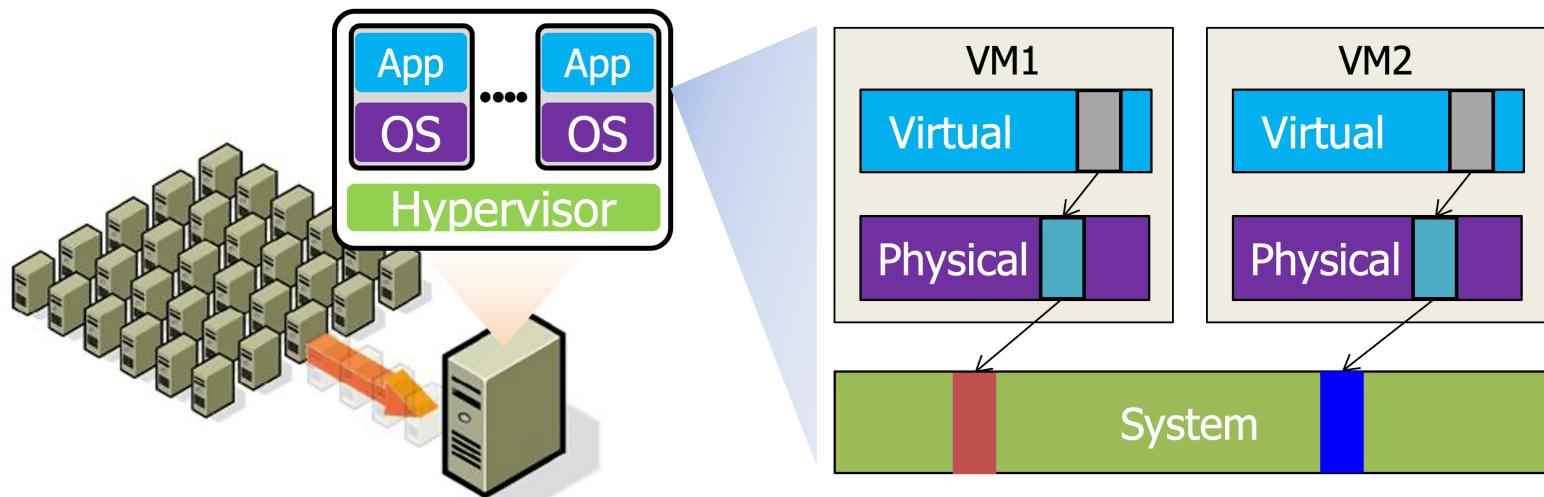
Virtualized CPU

- Application or VM (Guest OS) runs until:
 - Privileged instruction traps
 - System interrupts
 - Exceptions (page faults)



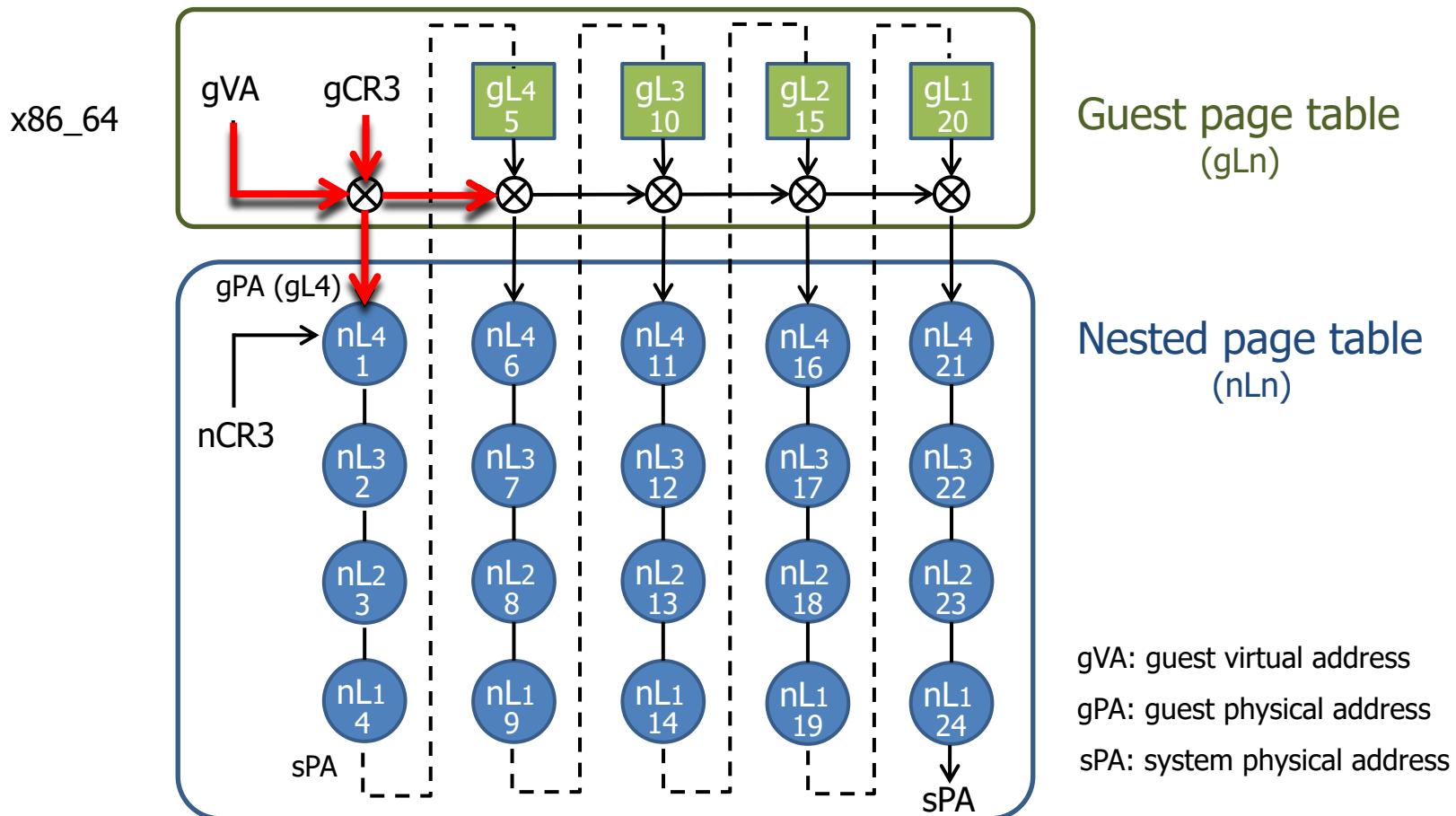
Virtualized Memory

- Virtualized Memory
 - 머신 메모리 (실제 메모리) 와는 무관하게 가상머신에게 할당된 메모리
 - 머신 메모리가 1G가 라도, 가상 머신에게는 100G도 할당 가능
 - 별도의 맵핑 테이블을 통해 제공
 - Virtual -> Physical -> Machine



Virtualized Memory

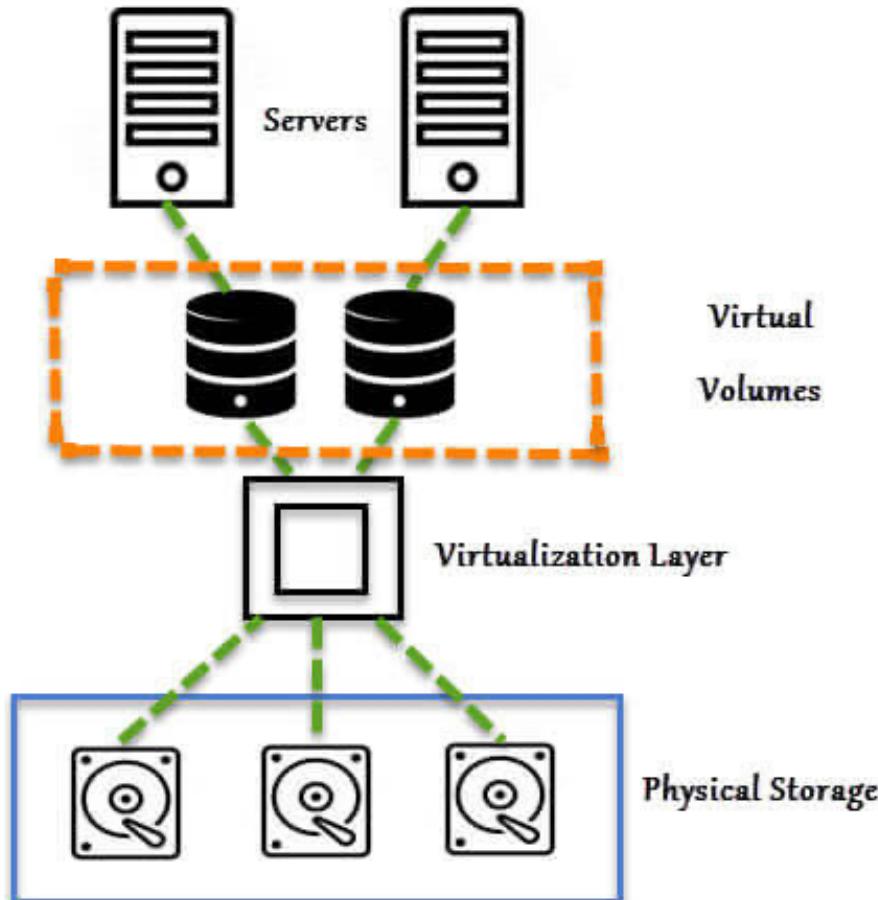
- Hardware-Assisted Page Walks



Storage Virtualization

- Storage
 - 가상화
 - 가상화 장소

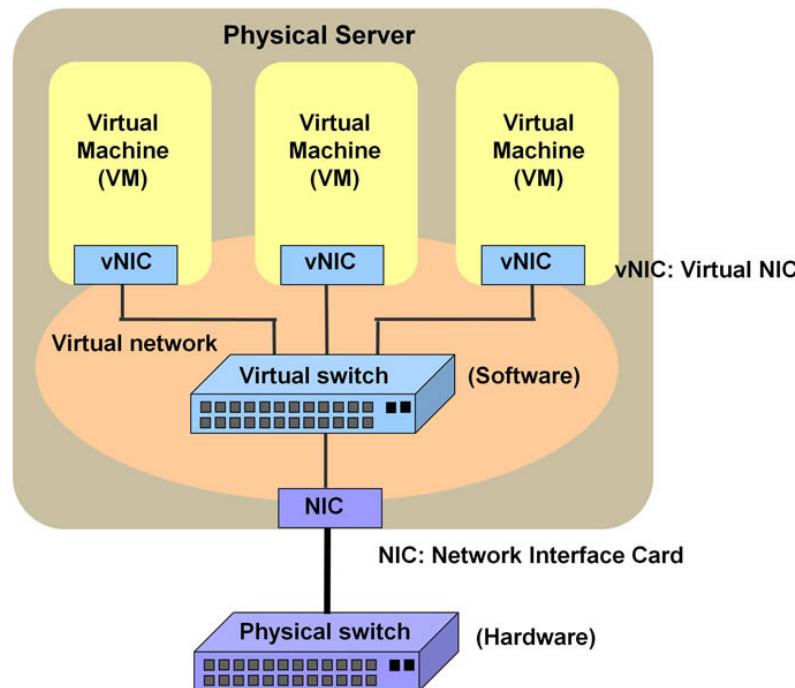
Storage Virtualization



! 때, 물리 저

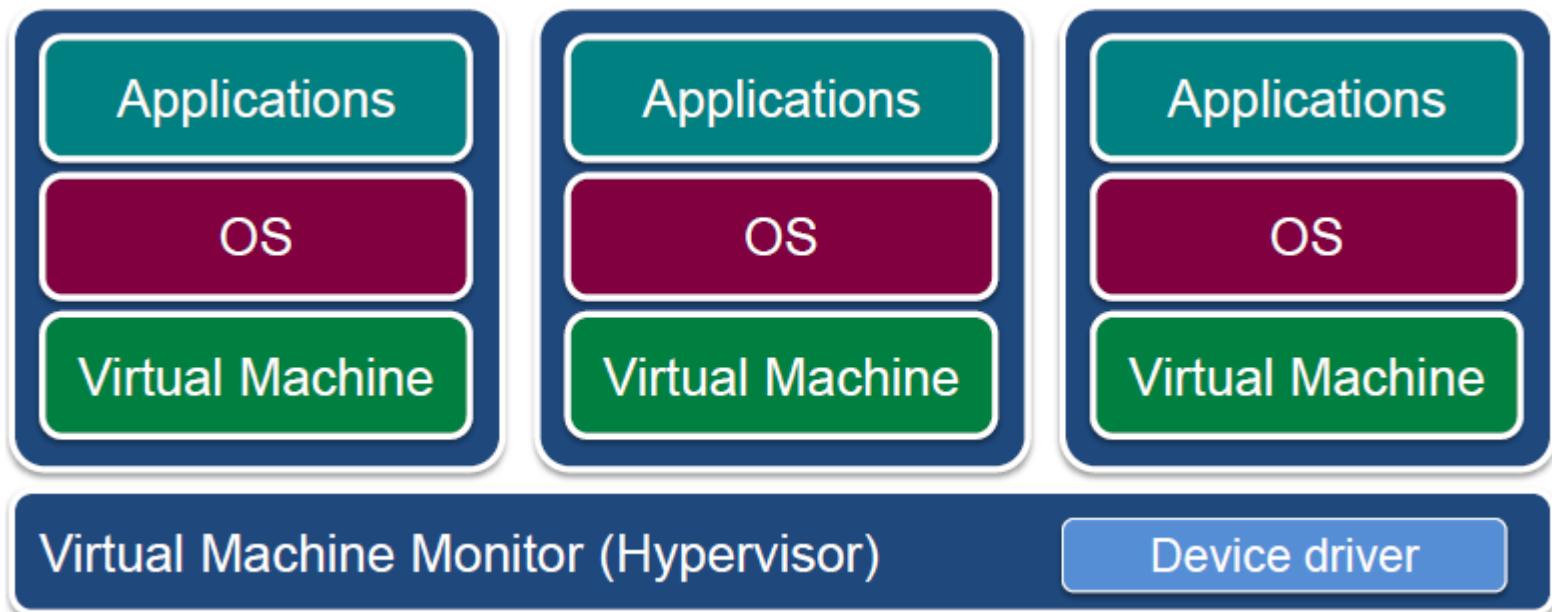
Network Virtualization

- **Virtualized Network Interface Card**
 - 가상 머신에게 vNIC를 할당
 - Software Switch를 통해 가상머신의 패킷을 외부로 수신/송신



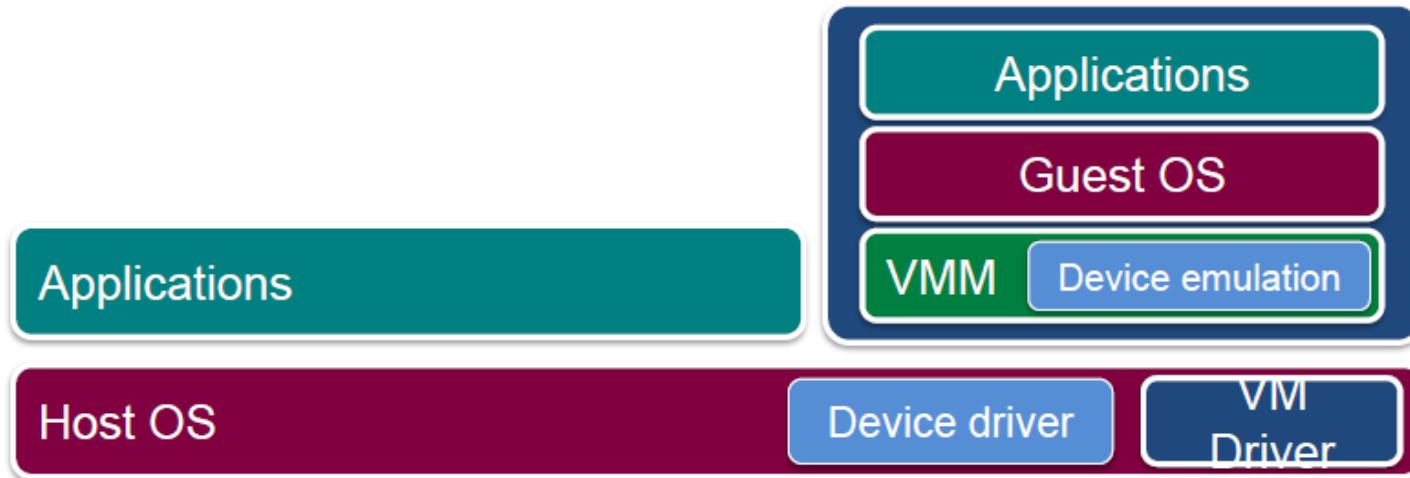
Native Virtual Machine

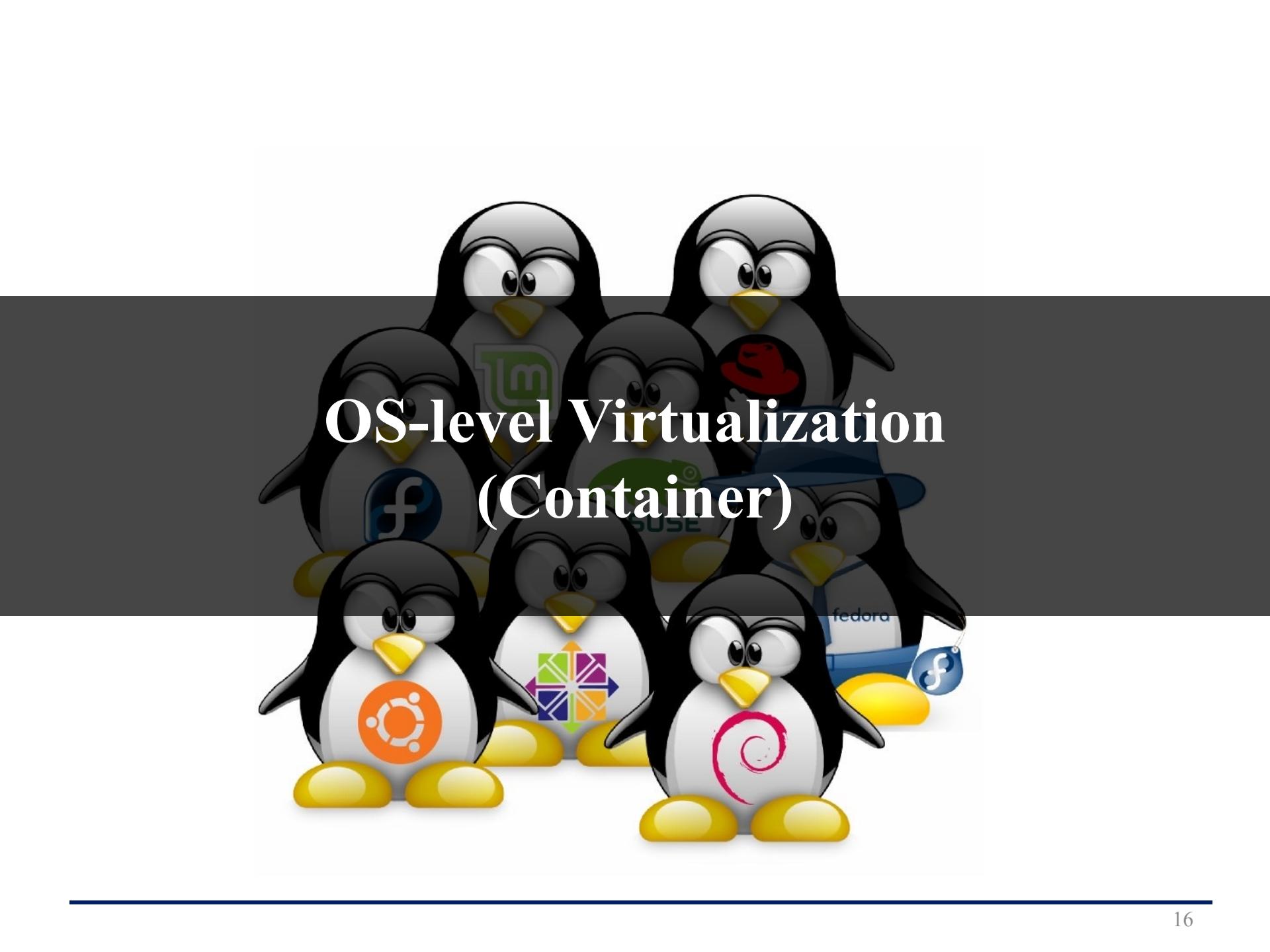
- Native VM (or Type 1 or Bare Metal)
 - Hypervisor가 하드웨어를 컨트롤
 - Hypervisor에는 각 하드웨어 맞는 디바이스 드라이버가 필요
 - Hypervisor는 가상 머신에게 자원을 할당해줌



Hosted Virtual Machine

- **Hosted VM**
 - 기존 OS의 기능을 활용하여 가상화를 제공
 - 기존 OS의 디바이스 드라이버를 활용함
 - 가상 머신은 기존 OS의 어플리케이션처럼 동작함



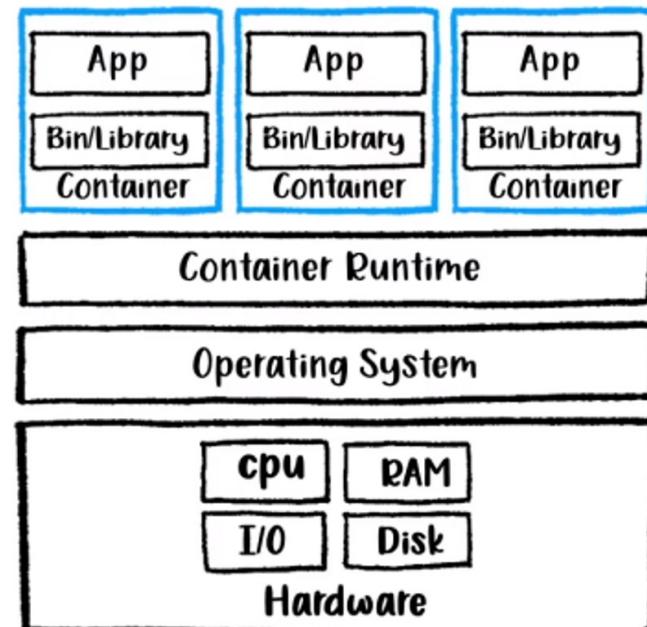


OS-level Virtualization (Container)

OS Level of Virtualization

- 격리된 환경

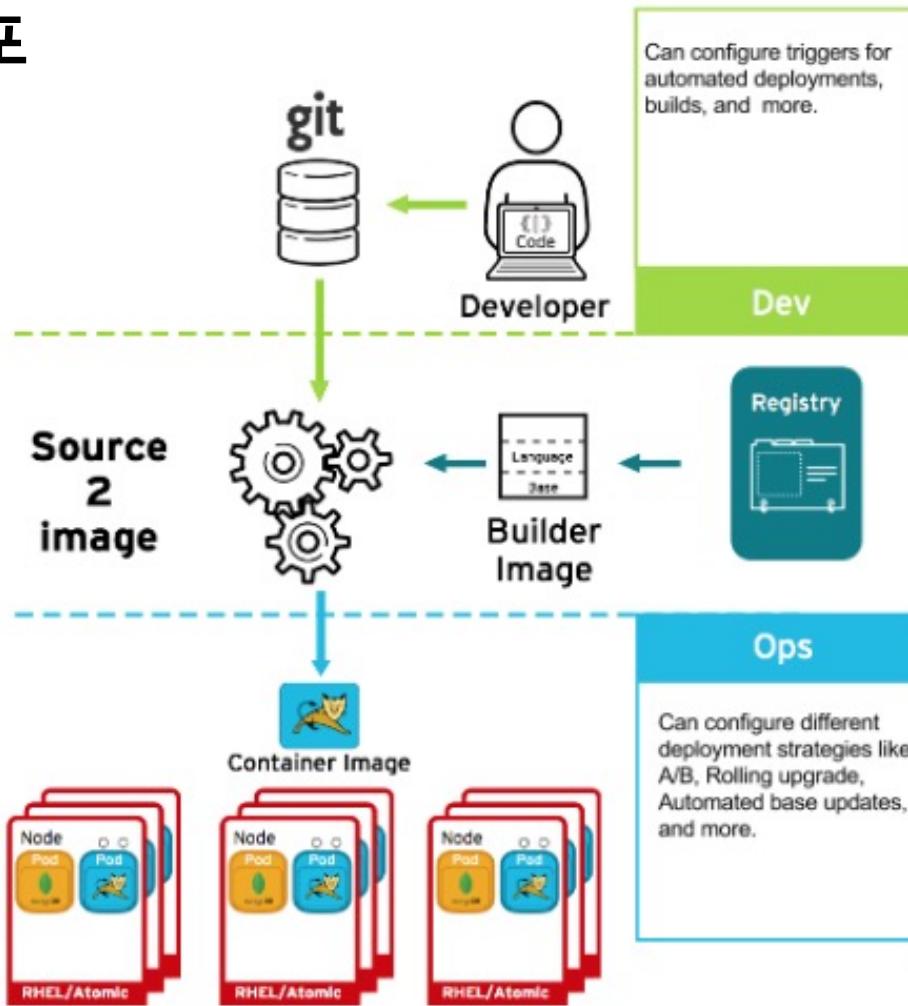
- 앱과 앱을 구동하는데 필요한 것들이 컨테이너에 보관됨
- 각 컨테이너끼리는 서로 영향을 주지 않음
- 안전한 테스트



Container

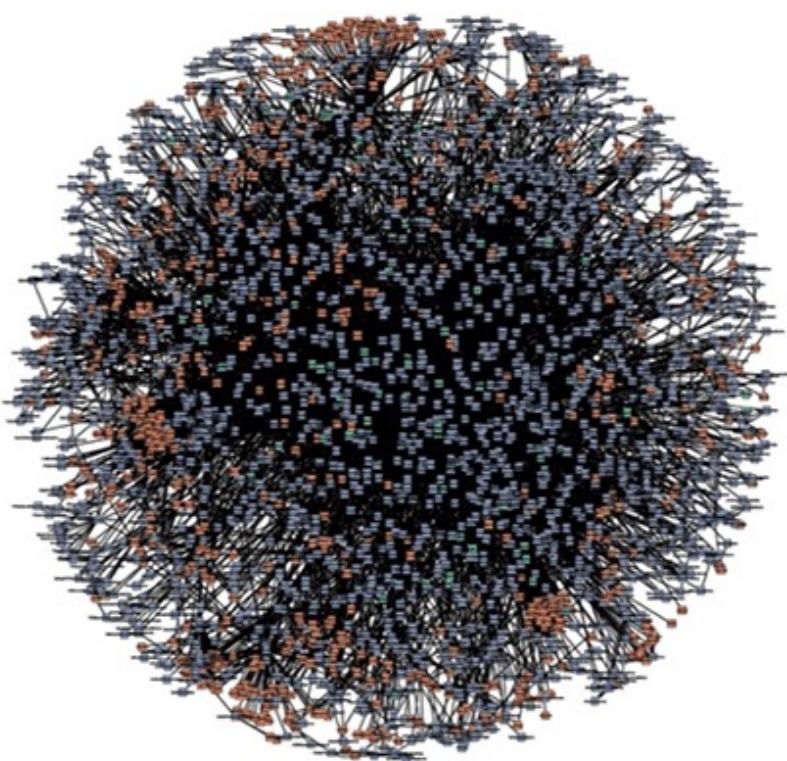
OS Level of Virtualization

- 손쉬운 배포



OS Level of Virtualization

- Micro Service Architecture (MSA) 적합



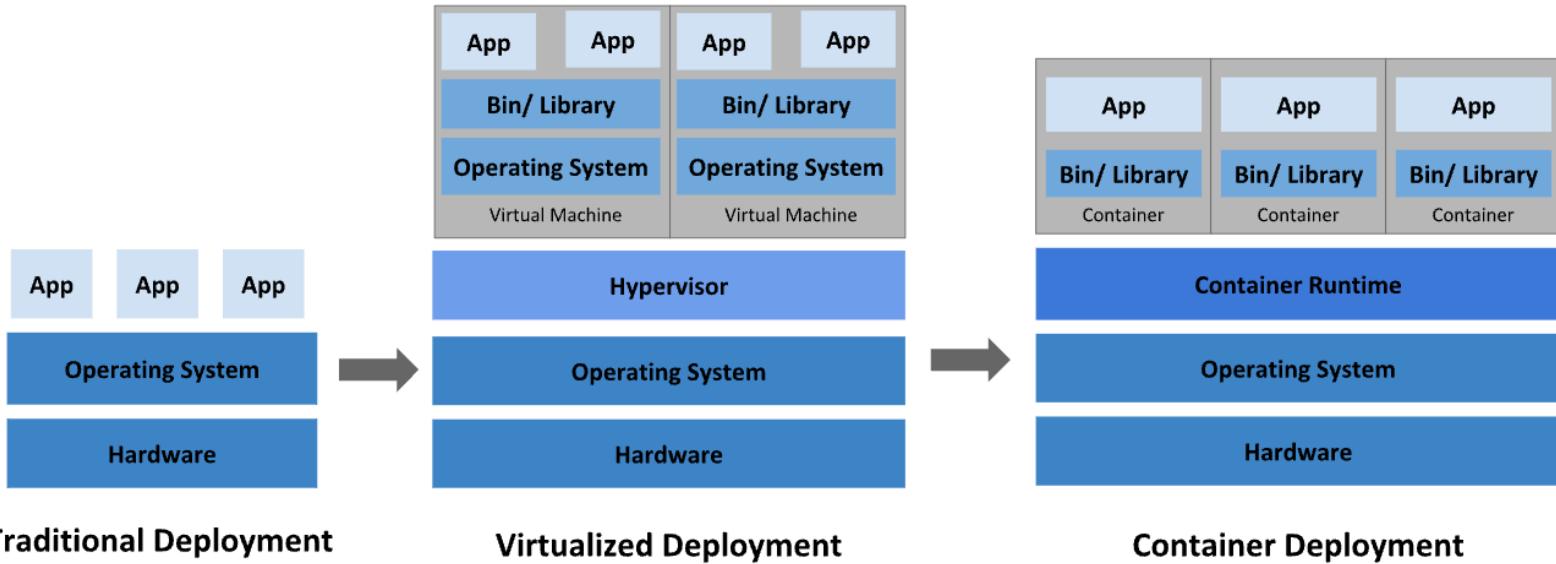
amazon.com®



OS Level of Virtualization

- **Virtual Machine (Hardware Virtualization)**
 - 각 하드웨어를 가상화하여 처리
 - 가상머신에는 별도의 OS가 동작함
- **Container (OS-level Virtualization)**
 - 가상머신 생성이 아닌 프로세스를 실행시키는 것과 동일
 - 하드웨어 가상화가 아닌 OS 내에서 가상화 기능을 제공함
 - 하나의 App을 실행하기 위한 각종 라이브러리 및 파일 등을 Container 단위로 만들어 가상화함

OS Level of Virtualization



- 컨테이너
 - 앱이 구동되는 환경(라이브러리, 설정 등)을 감싸서 실행할 수 있도록 하는 소프트웨어 패키지
- 컨테이너 런타임
 - 컨테이너를 다루는 도구 (e.g. Docker, CRI-O, containerd)

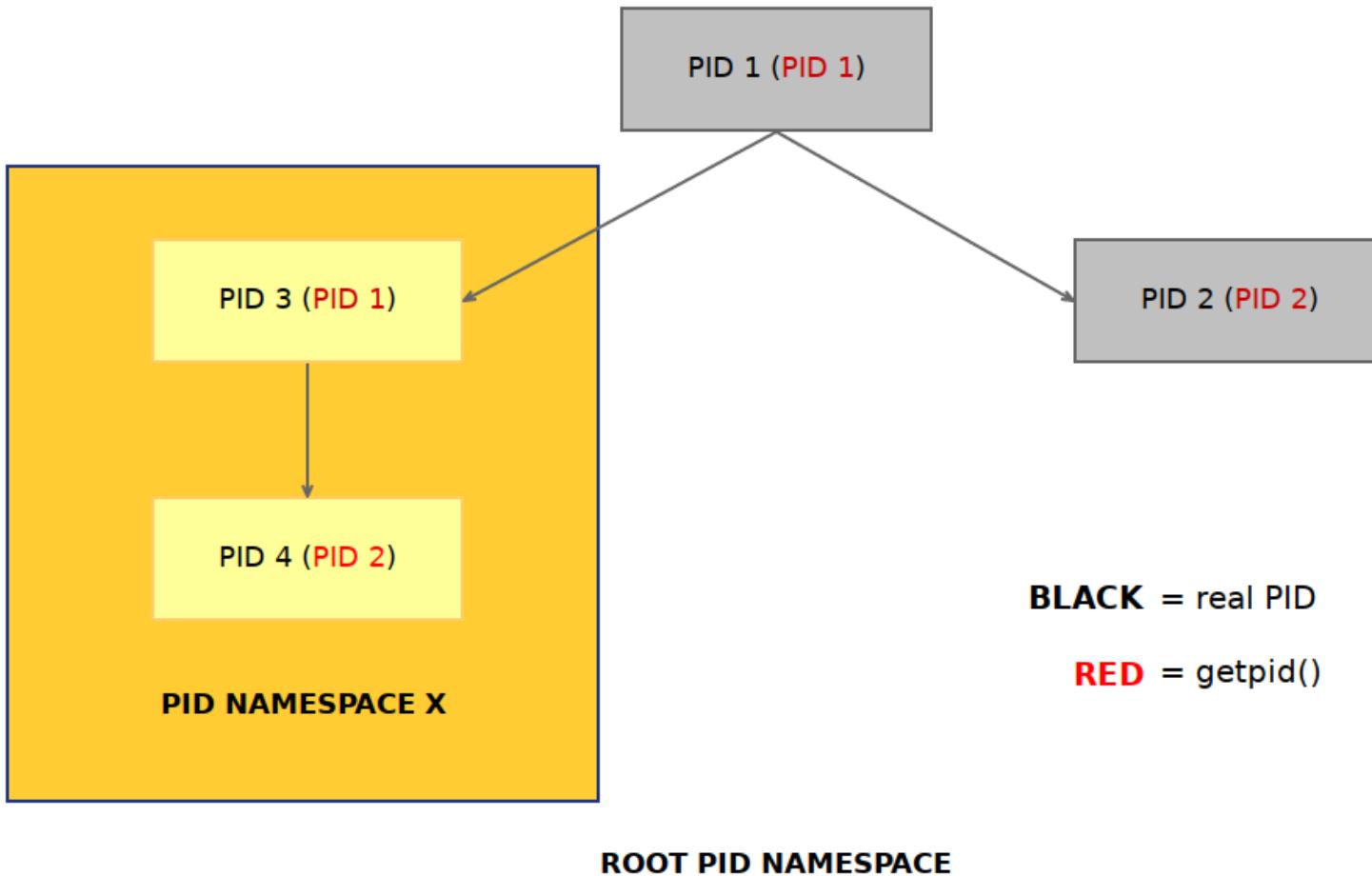
OS Level of Virtualization

- 장점
 - 배포 및 시작이 매우 빠름
 - 별도의 OS나 하드웨어 가상화가 불필요하여 성능이 우수
 - 각 Container 간의 격리 (Isolated) 만 제공하는 수준의 가상화
- 단점
 - OS에서 제공하는 가상화이므로 모든 Container는 동일한 OS 및 Kernel을 사용해야 함

Isolation

- Container
 - 기본적으로 가상화를 제공할 필요가 없고, 각 Container 간의 격리 (Isolation) 만 제공하면 됨
 - namespace, chroot, cgroups를 통해 제공
- Namespace
 - 특정 프로세스에 대해서 시스템 리소스를 논리적으로 격리
 - mnt : 독립적으로 파일시스템을 마운트
 - pid : 독립적으로 프로세스 공간을 할당
 - net : 독립적으로 네트워크 공간을 할당 (포트)
 - ipc : 독립적인 프로세스 간의 통신 경로 제공
 - uts : 독립적인 hostname 할당
 - user : 독립적인 user 할당

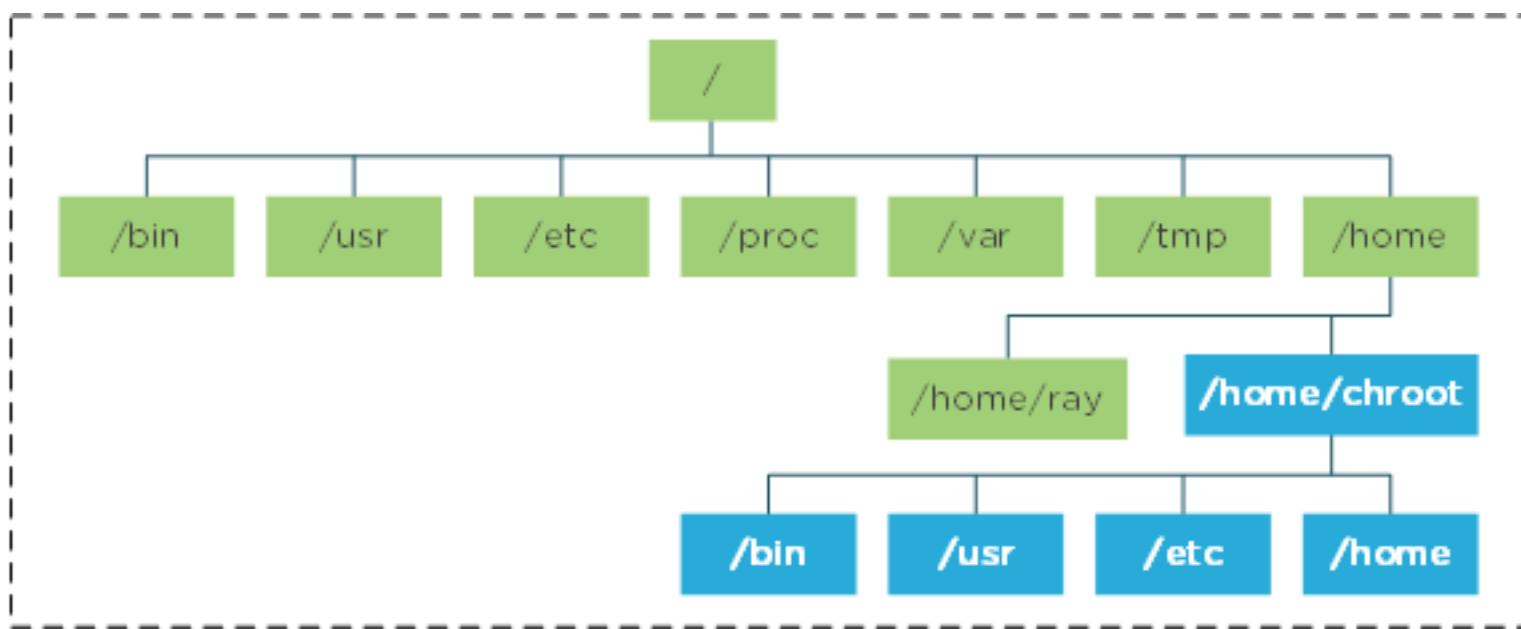
Isolation



Isolation

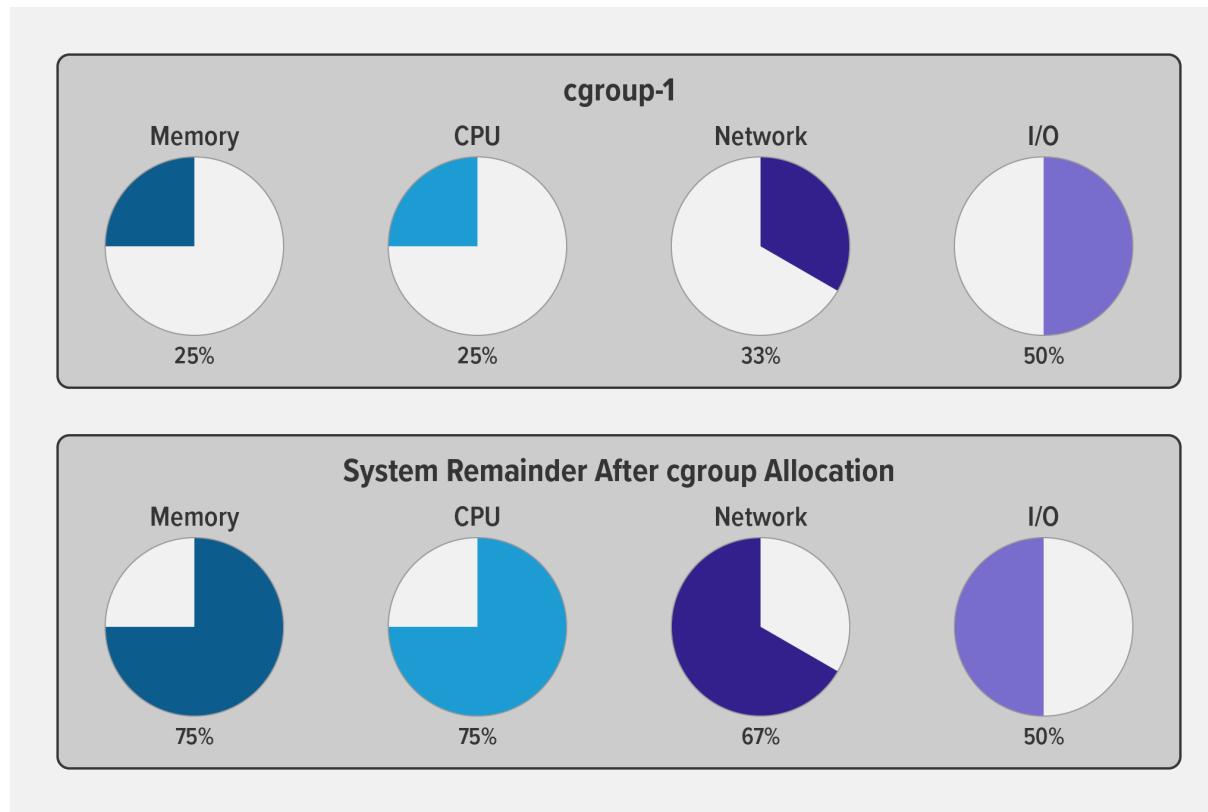
- **chroot**

- 가상의 root 디렉토리를 설정하여, 해당 디렉토리에서 벗어나지 못하도록 격리함
- 컨테이너가 다른 외부의 디렉토리를 접근하지 못하게 함



Isolation

- **cgroups**
 - 자원 (Resources)에 대해서 제어를 해줄 수 있음
 - 메모리, CPU, I/O, 네트워크, Device



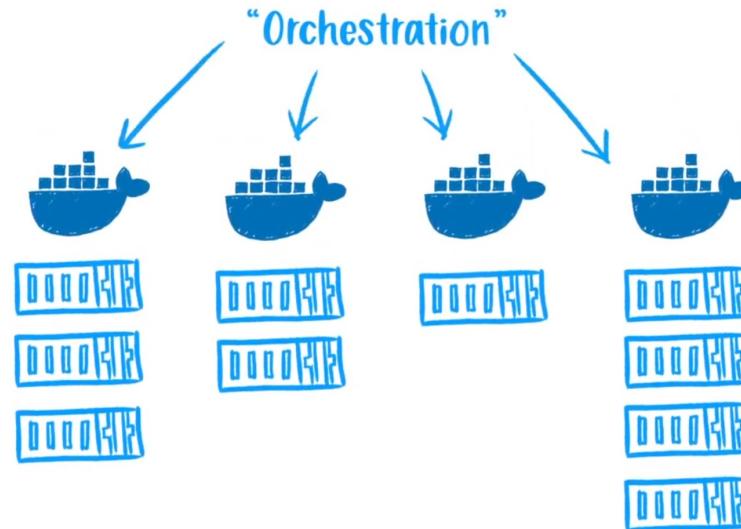
Why Container?

- 비교

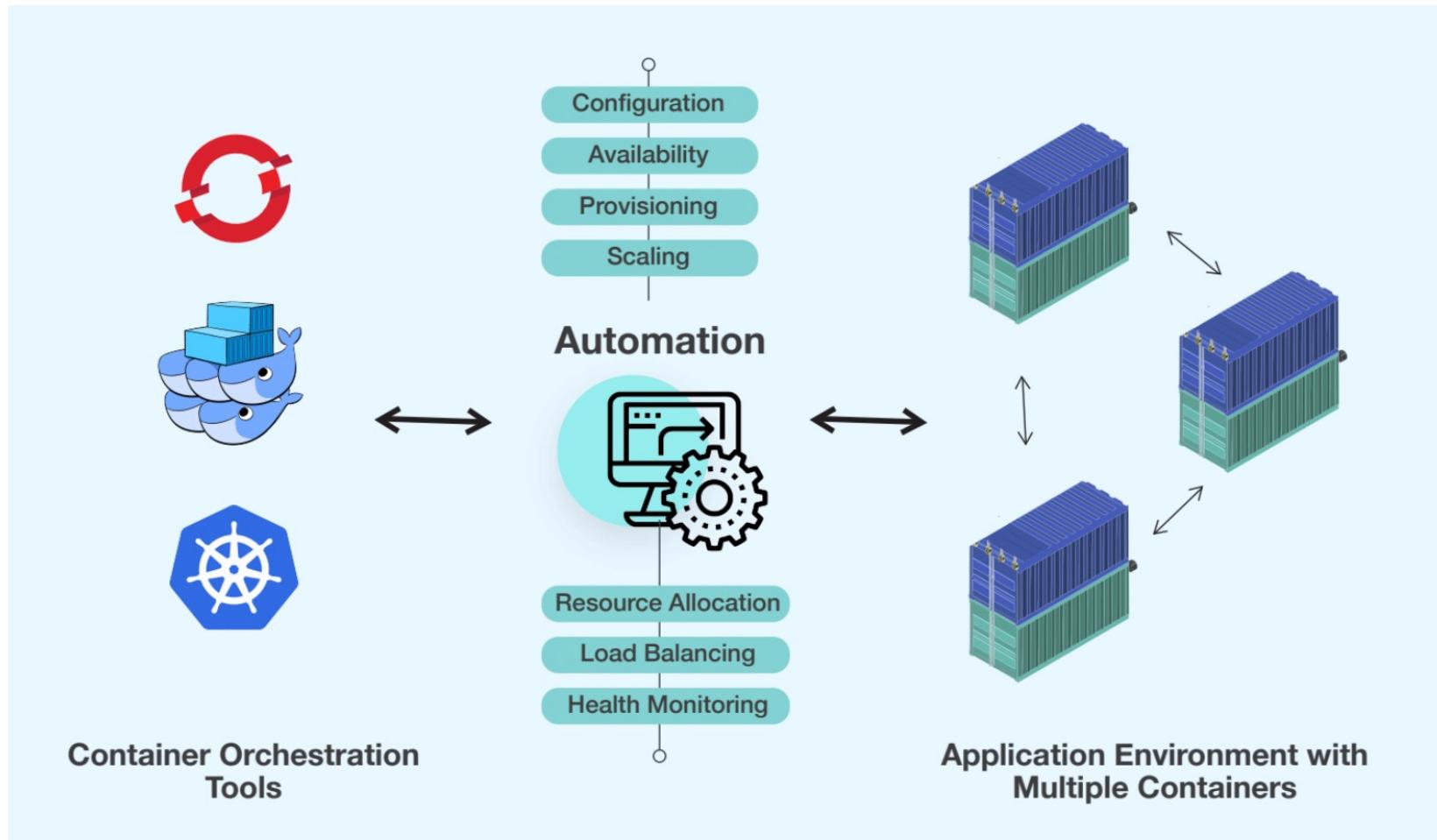
	가상머신	컨테이너
시작시간	몇 분	몇 초
이미지 크기	기가 단위	메가 단위
OS	가상머신을 위한 OS 필요	호스트 OS를 그대로 사용
성능	가상화 및 에뮬레이션을 위한 많은 오버헤드 발생	추가적인 오버헤드 없음
이식성	가상 이미지 변환 필요	컨테이너 이미지 그대로 이용

Container Orchestration

- 컨테이너는 각 기능 별 분리가 기본적인 철학
 - 하나의 큰 서비스를 하나의 컨테이너가 아닌 각 역할을 담당하는 다수의 컨테이너로 쪼갬 (MSA)
 - 이런 컨테이너를 사람이 설정하는 것은 매우 힘듦
 - 모든 것을 자동화!
 - 이를 위해 쿠버네티스와 같은 Orchestration tool이 필요

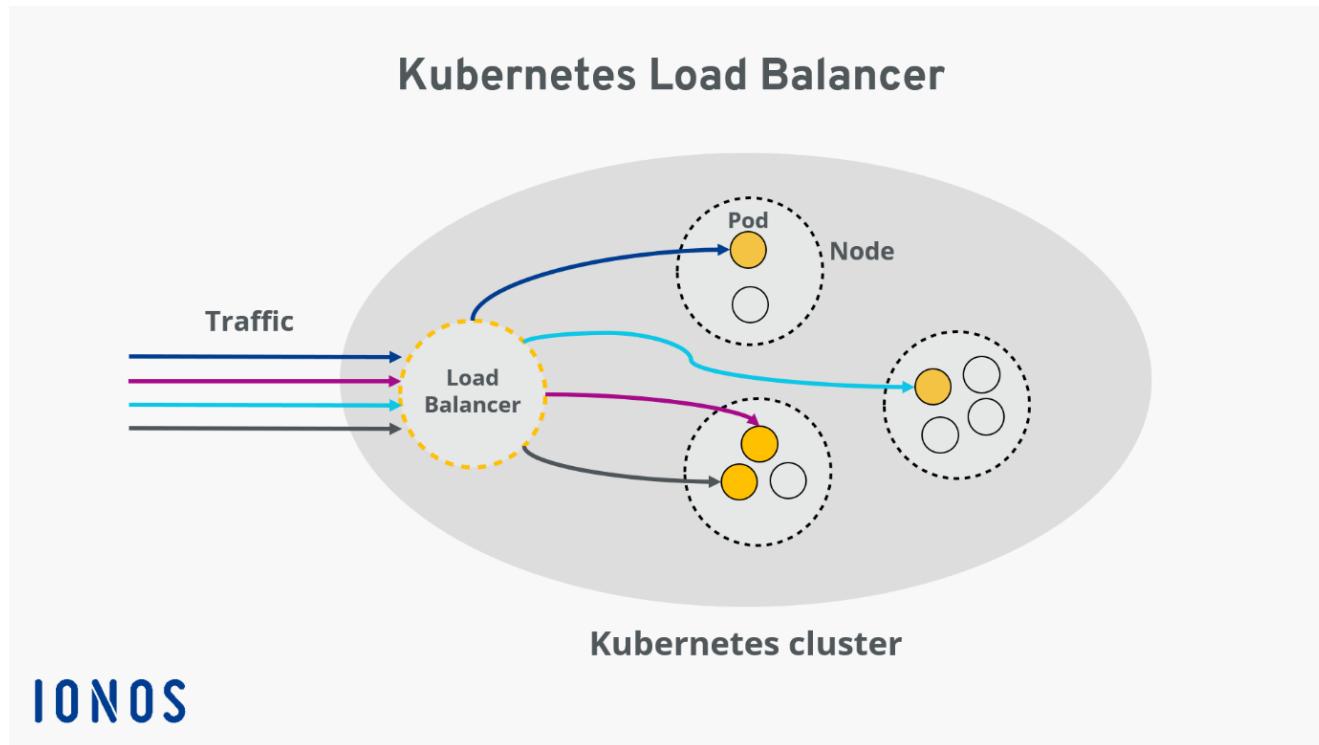


Container Orchestration



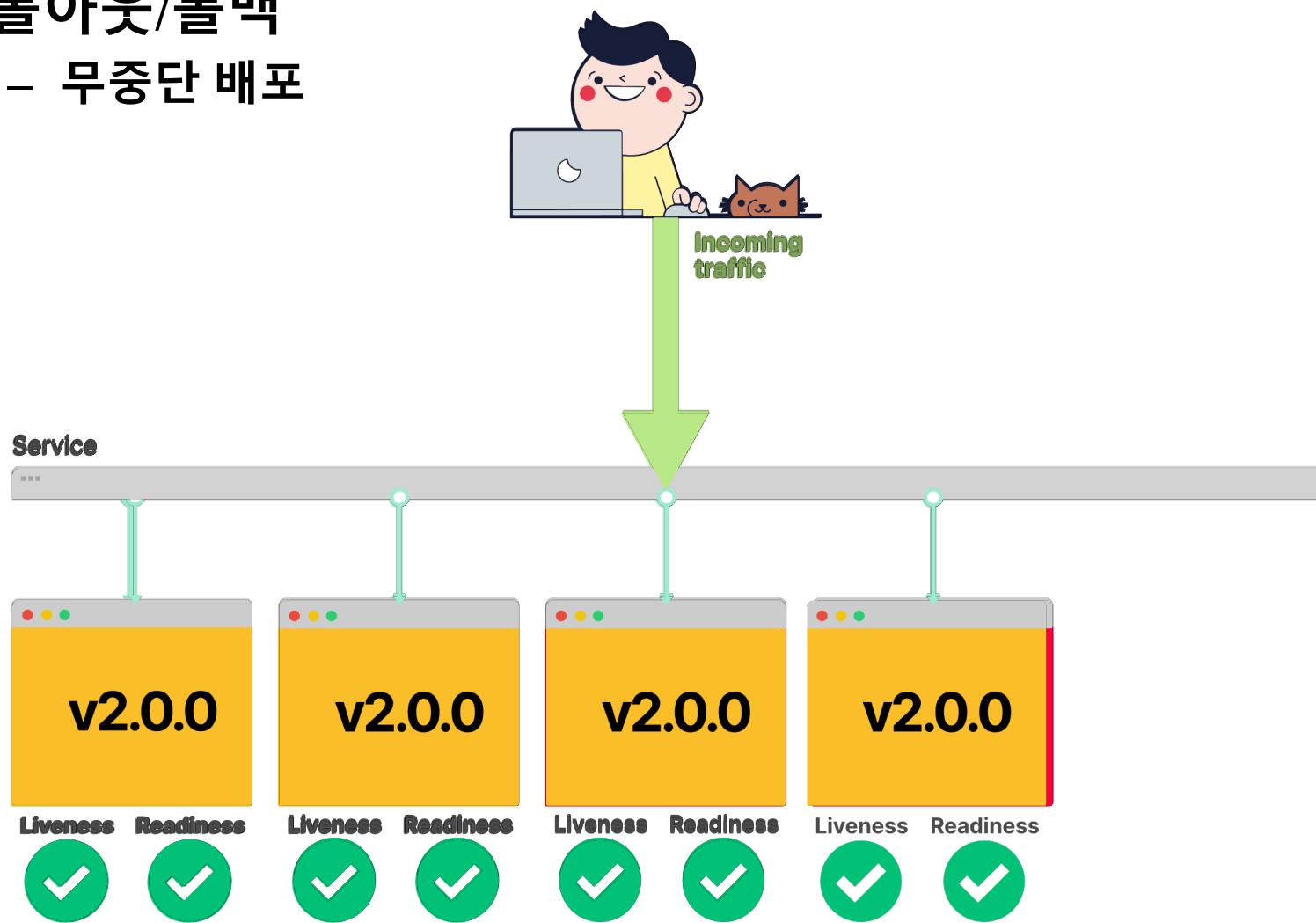
Container Orchestration

- 로드 밸런싱
 - 컨테이너 트래픽이 많아지면 자동으로 컨테이너를 추가 생성하고 트래픽을 분산 처리

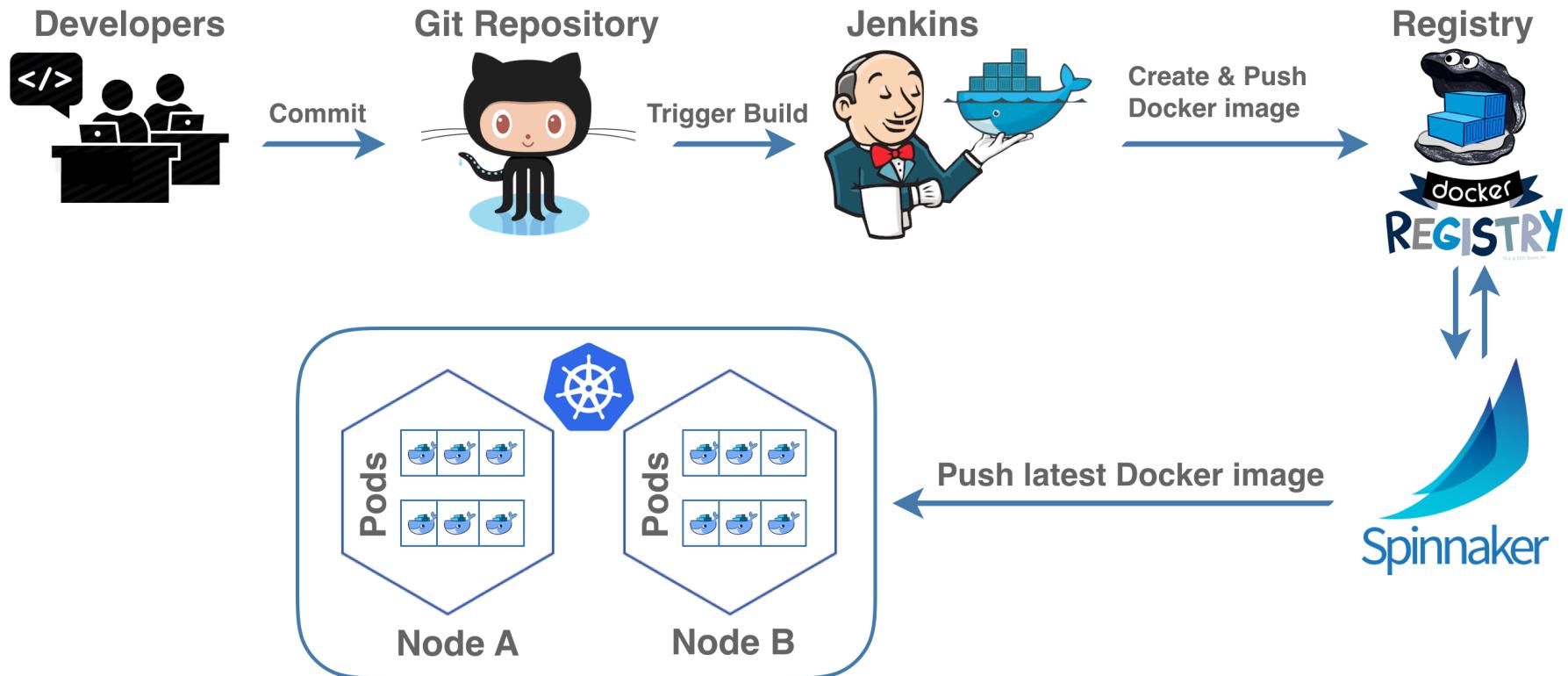


Container Orchestration

- 롤아웃/롤백
 - 무중단 배포



Container Orchestration



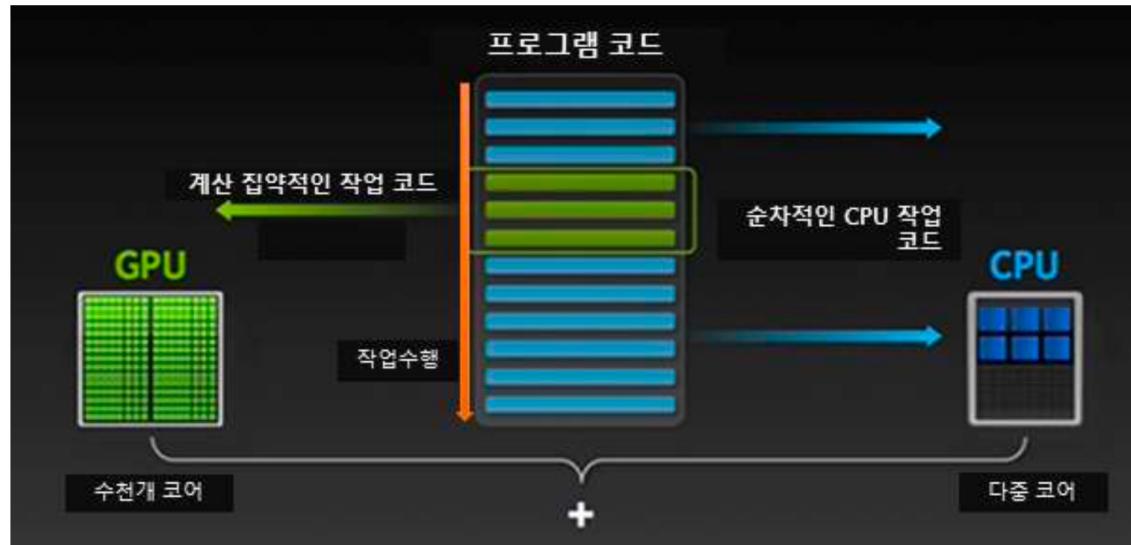
CI/CD for Kubernetes Cluster with Jenkins and Spinnaker

GPU Virtualization

GPU Virtualization

- GPU

- GPU는 많은 코어 (1000개 이상)를 가지고 있어 병렬처리가 매우 우수함
- CPU와 비교하면 단위 코어 당 가성비가 매우 높음
- AI와 같은 많은 컴퓨팅 파워를 요구하는 곳에 매우 적합함
- 클라우드에서 AI와 같은 워크로드에 대한 수요가 매우 증가
- 비싼 GPU를 어떻게 효율적으로 나눠 (Share) 쓸 수 있을까?

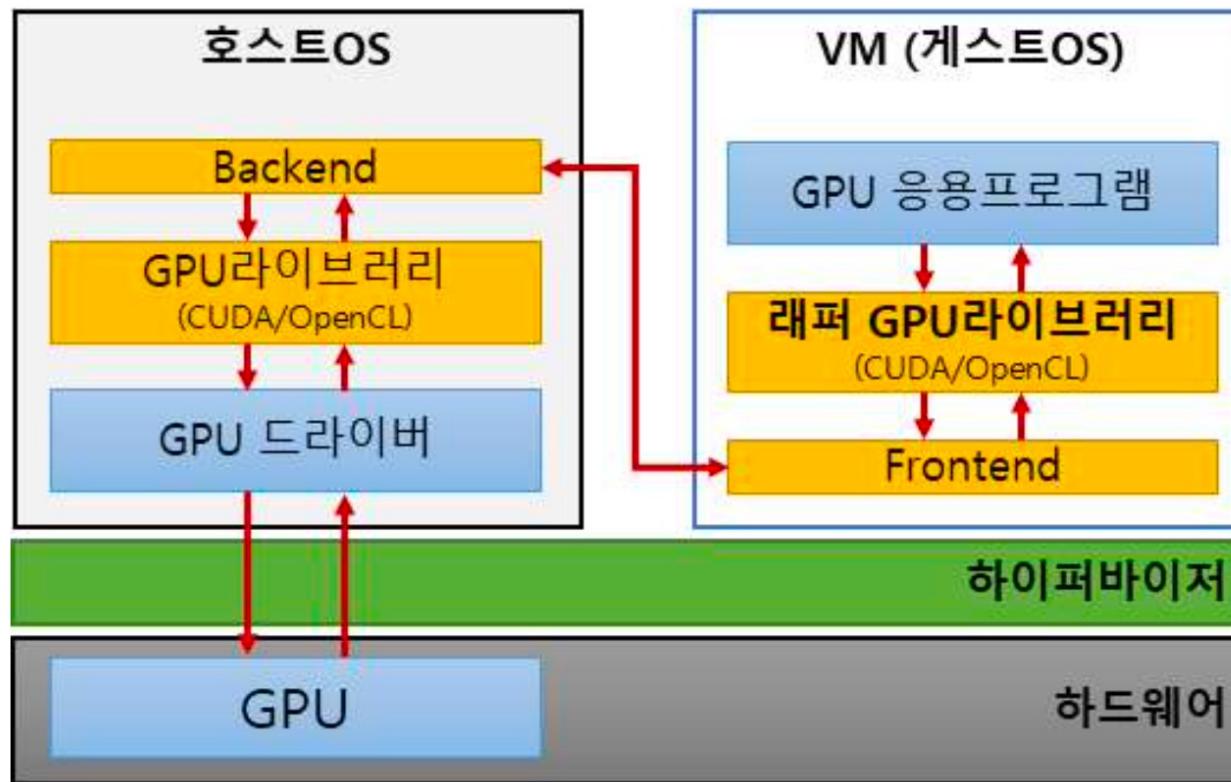


GPU Virtualization

- GPU 가상화
 - GPU의 활용률은 15%에 불과한 경우가 많음
 - GPU 제조업체의 지원이 없어서 그 동안은 가상화가 매우 어려움
 - 이를 극복하기 위한 몇 가지 대안들이 제시됨

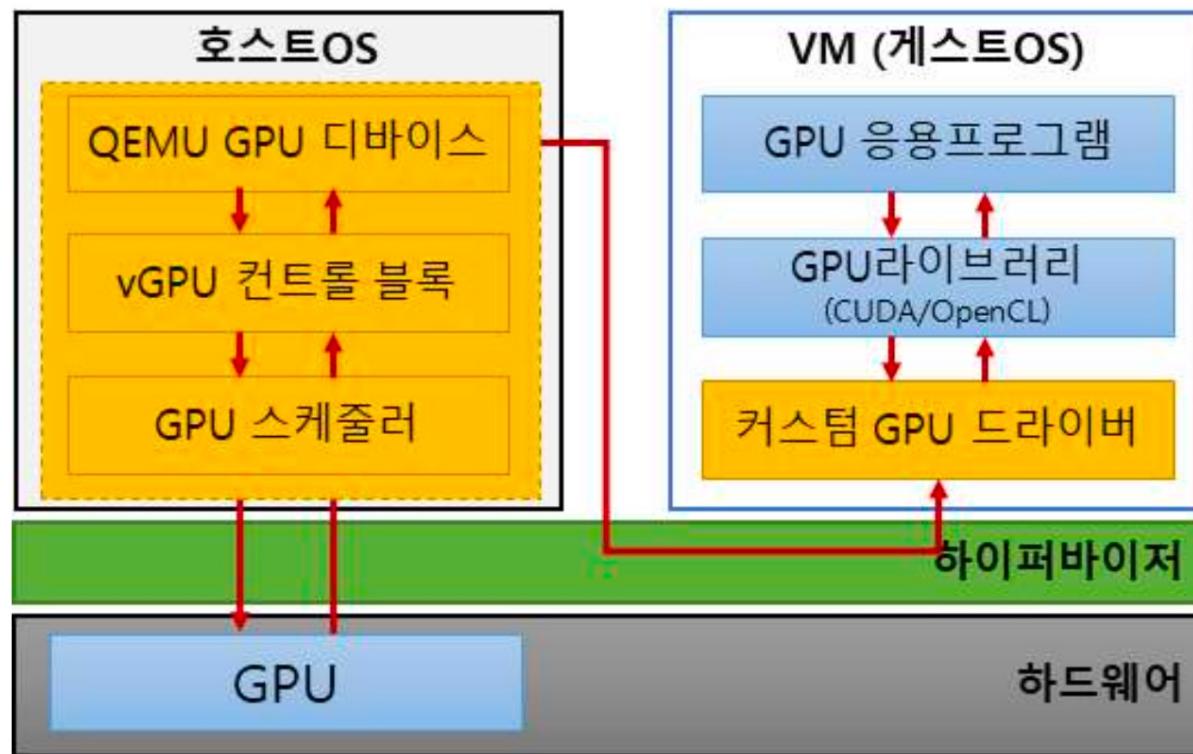
GPU Virtualization

- API Remoting
 - 원격의 가상머신에서 GPU 가진 머신에 요청을 보냄



GPU Virtualization

- 드라이버 가상화
 - 커스텀 GPU 드라이버는 공유메모리를 통해서 QEMU GPU 디바이스에게 요청을 보냄



GPU Virtualization

- 하드웨어 지원 GPU 가상화
 - GPU 벤더가 가상화를 지원함
 - 가장 성능이 우수함

