

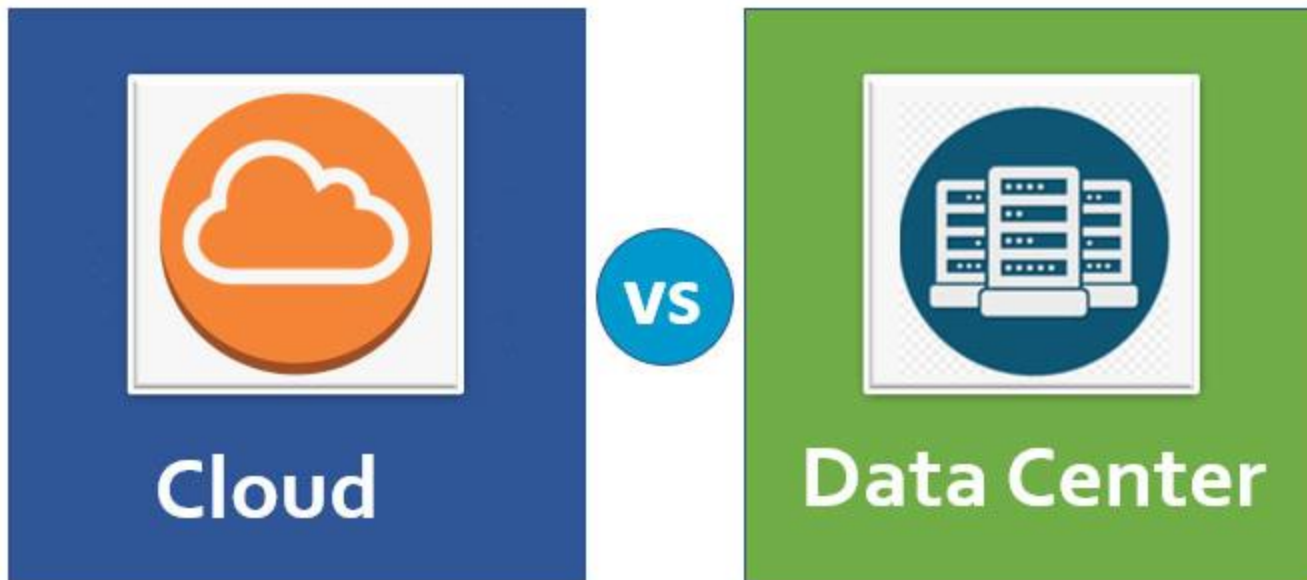
# Cloud Computing 소개

Data Center vs  
Cloud

Virtualization vs  
Container

Private Cloud vs  
Public Cloud

# 1-Data Center vs Cloud



# Data Center

- Data Center란?

- Application Server를 Hosting하는 실제 시설을 말한다

- 컴퓨팅 시스템을 위한 하드웨어
    - Networking Device
    - 전원 공급장치
    - 전기 시스템
    - backup용 발전기
    - 환경 제어 장치(냉각장치, 팬 등)
    - 운영 인력

- 데이터 센터의 문제점

- 엄청난 운영 비용
    - 건물 유지 비용, 서버 구매 비용, 초기 구축 비용, 유지 보수 비용
  - 느린 구축 시간
    - 사용자의 수요에 빠르게 대처하기 힘들다
    - 장애간 난 장비를 교체하는 시간이 느리다



- Cloud란?

- Cloud Computing은 IT Resource를 Internet을 통해 On Demand로 제공하고 사용한 만큼 비용(Pay as you go.)을 지불하는 것이다
- 세탁기 vs 세탁소
- 지방 장기 출장 때 이용할 숙박 시설 선택
  - 직접 거주할 집을 건축
    - 집 설계, 부지 탐색, 계약, 등기, 업체 선정, 건축 등
    - 내가 원하는 대로 집을 지을 수 있지만, 초기 비용이 많이 들고, 건축 기간도 오랜 걸리고, 상황 변경(갑작스런 출장 종료, 인원 변경)에 쉽게 대처도 힘들고, 유지 보수(인터넷 업체 변경, 에어컨 수리, 형광등 교체 등)도 직접해야 한다
  - 호텔을 이용한다
    - 단순히 Check-In 및 Check-Out만 하고 이용한 만큼 숙박료를 지급하면 된다(On Demand)
    - 적은 거주 비용, 곧장 이용 가능, 유연하게 이용도 가능하고, 유지 보수를 내가 할 필요가 없다

- Cloud 장점

- 자원 비용을 가변 비용으로 대체
  - 데이터 센터 구축 비용, 서버 구매 비용 -> 운영비
  - 엄청나게 많이 소요되는 초기 구축 비용 대신, 사용한 만큼만 비용 지불
- 규모의 경제의 혜택
  - 한 개를 구매하는 것보다 1000개를 구매하는 것이 단가가 낮음
  - Cloud 규모 경제로 인한 이득 획득
  - Cloud의 모든 고객과 공동 구매하는 효과
- 민첩성 및 속도 개선
  - 몇 번의 클릭으로 공장 자원 확보
  - 개발 비용 절감
- 데이터 센터 운영 및 유지 관리에 비용 투자 필요 없음
  - 인프라 관리가 아닌 비즈니스에 자원 집중
- 빠른 확장성
  - 몇 번의 클릭으로 글로벌로 서비스 가능

# Data Center vs Cloud

## Traditional Data Center Model

- Static and inflexible
- Antiquated tools
- Capital intensive
- Labor intensive



## Cloud Model

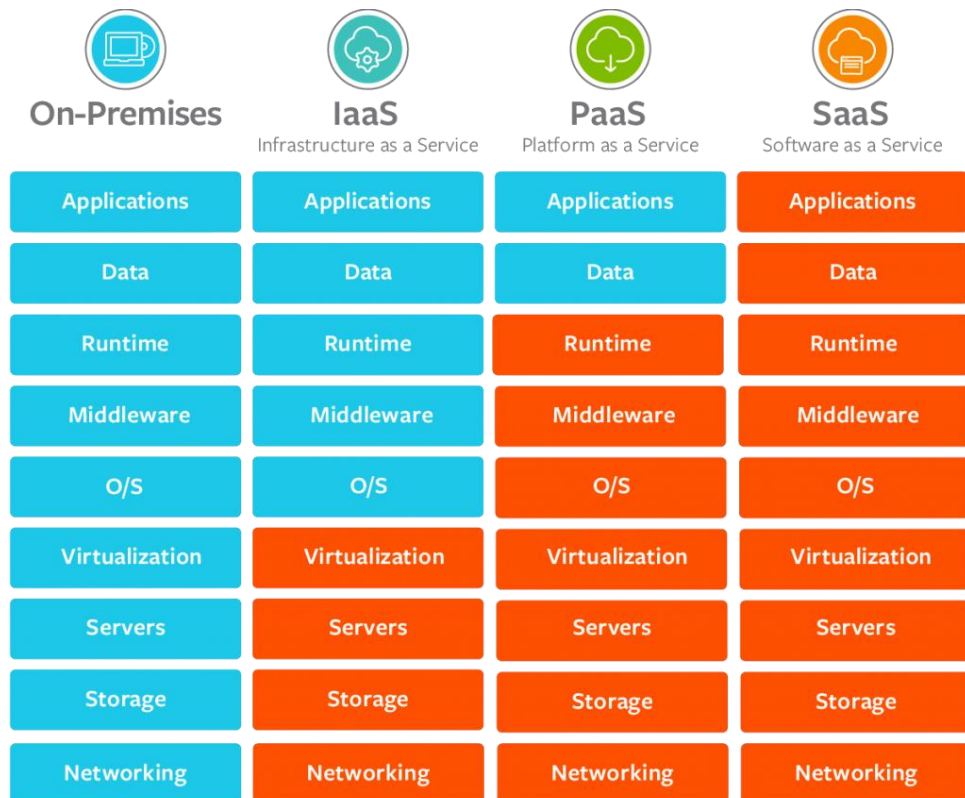
- Dynamic and flexible
- Modern tools
- Pay-as-you-go transparency
- Automated and efficient



# Cloud

- Cloud Computing 이점

- 상대적으로 초기 투자 비용이 적음
- 수요에 대한 빠른 대처 가능
- 불확실한 수요 예측에서 오는 손해가 적음
- 규모의 경제 혜택을 볼 수 있음
- 제품 개발에 집중할 수 있음
- 유지 보수가 쉬움

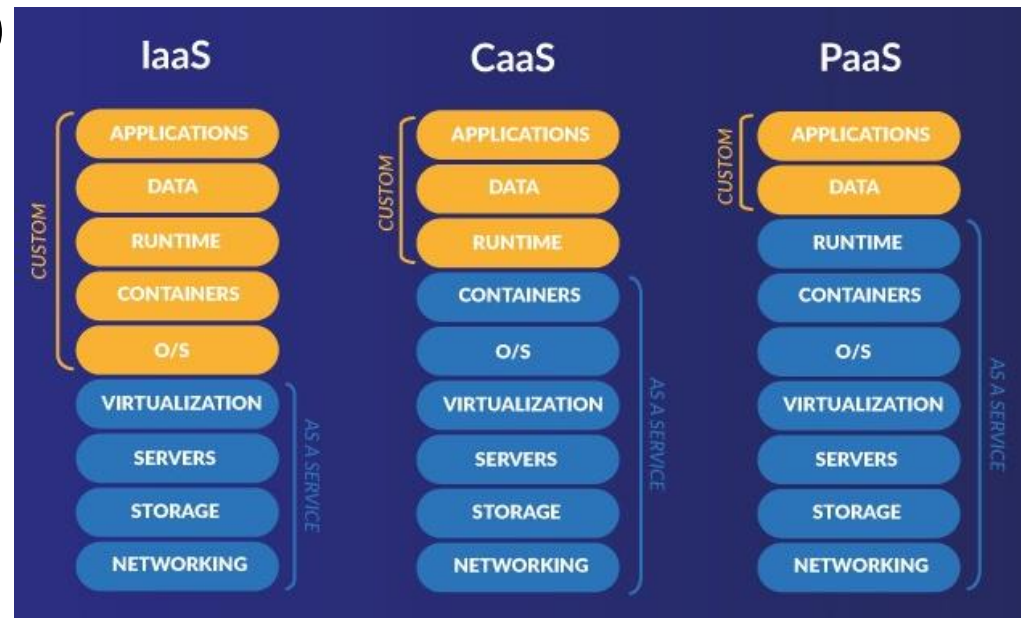
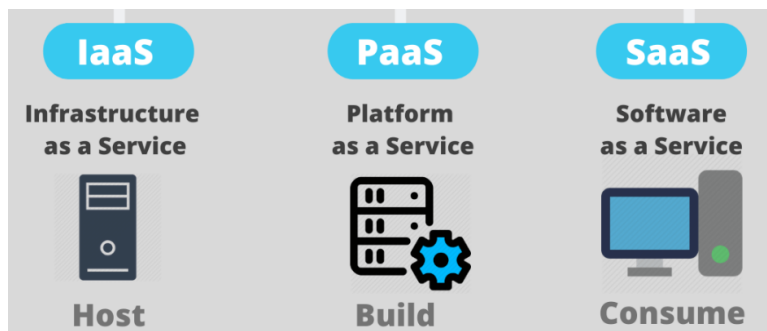




# Cloud 종류

- Cloud Computing **모델**

- Cloud에서 **무엇을 제공하느냐에** 따라 구분하는 것이다
- 모든 것이 구매하여 자기 것으로 소유하는 것이 아니라, On Demand로 빌려서 사용하는 Service이다
- IaaS(**I**nfrastucture as a Service)
- PaaS(**P**latform as a Service)
- SaaS(**S**oftware as a Service)
- CaaS(**C**ontainer as a Service)
- DaaS(**D**esktop as a Service)
- FaaS(**F**unction as a Service)



- Cloud Computing **모델-IaaS**

- IaaS는 **Infrastructure**만 제공하는 것이다
  - VM, Storage, Virtual Network 등등
  - Amazon EC2, Azure Virtual Machines, Google Compute Engine, Linode, DigitalOcean
  - 그냥 하드웨어와 OS가 준비된 가상 컴퓨터(VM)를 잠시 빌리는 것이다
- 휴가지에서 지내게 되는 팬션의 **주방 시설**
  - 음식을 해서 먹을 수 있는 환경(가스, 싱크대, 수도 등)만 제공
  - 그릇, 요리 재료, 요리하기는 투숙객이 모두 준비한다



- Cloud Computing **모델-PaaS**

- PaaS는 인프라가 있는 상태에서 **Platform**만 제공하는 것이다
  - Database, App service 등
  - AWS Elastic Beanstalk, Engine Yard, Google Cloud, Heroku, IBM Cloud, Mendix aPaaS(Mendix Application Platform as a Service), Microsoft Azure Pipelines, Red Hat OpenShift, VMware Cloud Foundry, Wasabi Cloud Storage
- 개발자가 프로그램을 개발하려면 **프로그램 개발 프로그램 및 실행 환경이 컴퓨터에 미리 설치되어** 있어야 하는 것과 같다
  - VSCode, Container Run Time
- 개발자는 그냥 Code만 만들어서 돌리기만 하면 된다
- **일을 할 수 있는 모든 환경과 구성품이 준비되어** 있는 것이다
- 팬션의 주방 시설에다 **그릇(주방기기) 및 요리 재료**까지 준비된 것이다
  - 투숙객은 몸소 물을 데우거나 후라이팬에 식재료를 넣어서 음식을 만들기만 하면 된다

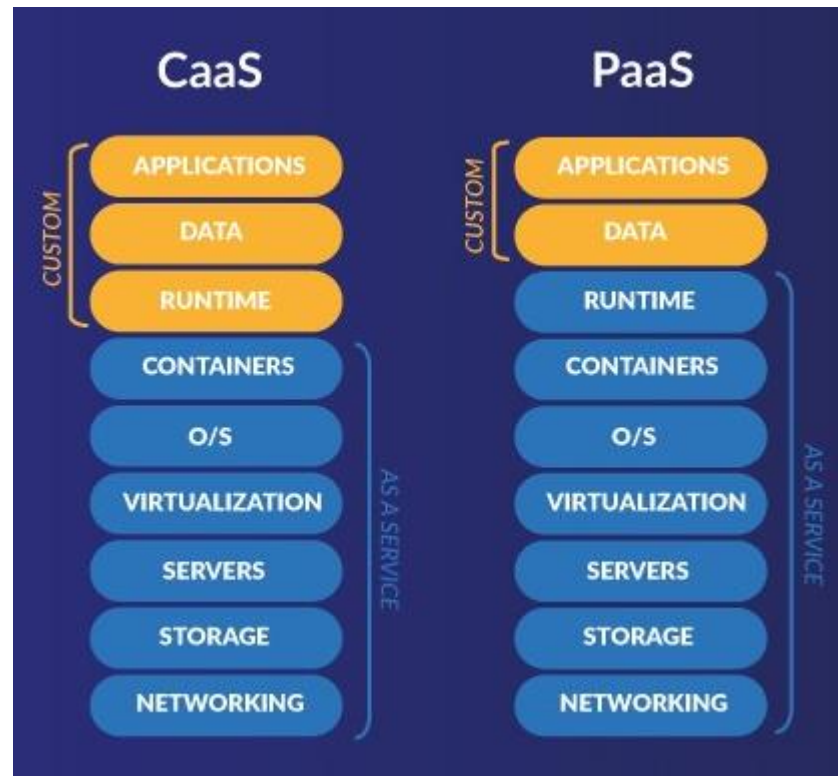
- Cloud Computing **모델-SaaS**

- SaaS는 **이미 설치된 소프트웨어**를 사용하도록 준비해둔 것이다
- 서비스 자체를 제공하고 다른 설정 없이 그냥 서비스를 사용하기만 하며 된다
  - Microsoft 365, Dropbox, Google Drive
  - Salesforce, Microsoft, Adobe Creative Cloud, Google Workspace
- 호텔에 투숙한 고객은 아침에 식당 및 Cafeteria에 가서 **이미 요리된 음식을 주문해서 먹기만 하면 된다**
  - 아침 식사를 준비할 필요가 없이 비용만 지불하고 먹고 종업원의 친절한 서비스를 받기만 하면 된다

# Cloud 종류

- Cloud Computing **모델-CaaS**

- CaaS는 Container 운영 및 서비스를 Cloud에서 제공하는 것이다
  - ACI(ECS), AKS(EKS)
  - Amazon ECS, Amazon Fargate, Azure Container Instance, Google Cloud Engine, Oracle Container Service



- Cloud Computing **모델-DaaS**

- DaaS는 cloud desktop service이다
  - Microsoft Windows Virtual Desktop
  - Amazon WorkSpaces
  - Citrix Managed Desktops
- 사용자는 최신의 OS를 사용할 수 있으면 보안 및 업그레이드에 대한 부담이 없다
- DaaS는 VDI의 모든 장점을 제공하면서, 사내 인프라가 아닌 Cloud Infra에서 구독 기반으로 운영된다



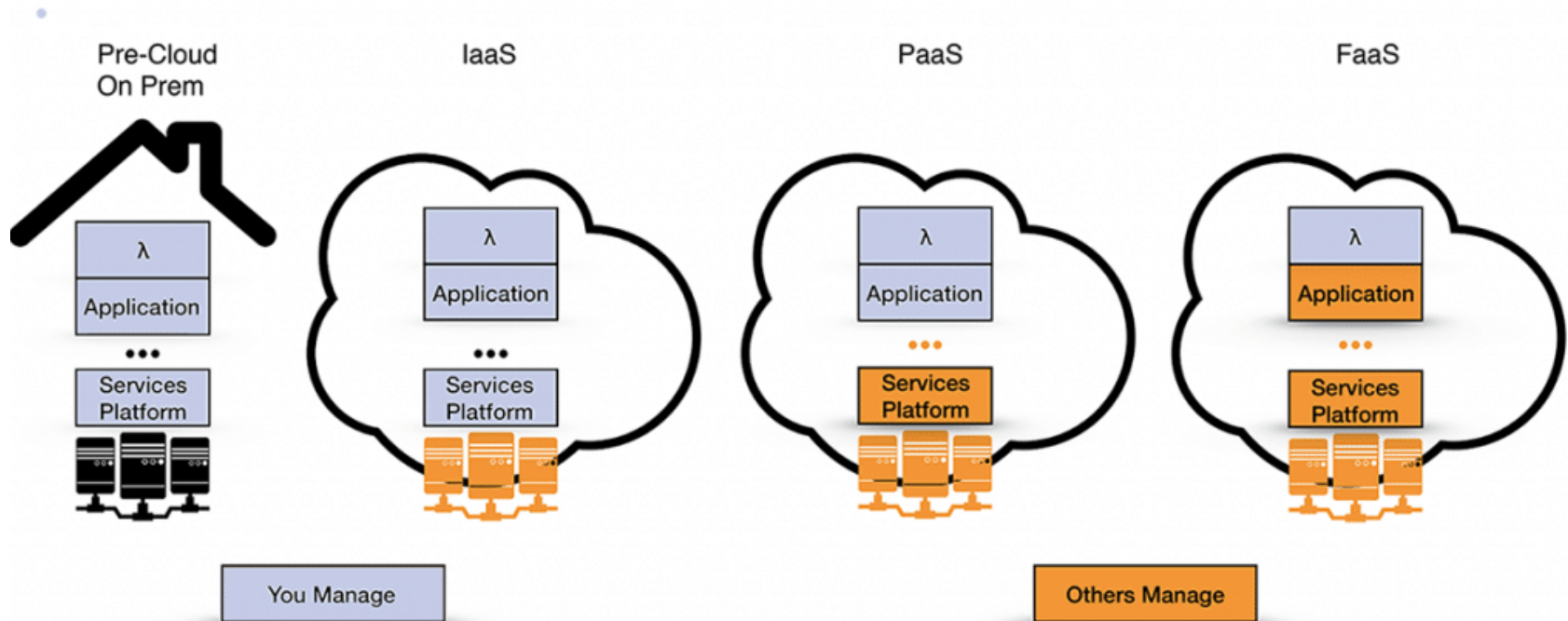
- Cloud Computing **모델-FaaS**

- FaaS는 Serverless Computing을 제공한다
  - AWS Lambda, Google Cloud Functions, Azure Cloud Functions, Cloudflare Workers
- FaaS 사용자는 서버/인프라를 직접 운영하지 않고 특정한 application function 또는 component를 실행할 수 있다
- FaaS를 사용하면 사용자는 단지 code만 입력하고 시간 및 실행 횟수에 따라 비용만 지급한다
  - CaaS를 사용하는 사용자는 Container를 구성하고 관리해야 해야 하지만, FaaS를 사용하는 사용자는 Infra에 전혀 신경 쓰지 않는다
- FaaS에서 제공하는 일반적인 기능
  - Authentication services, Database services, File storage, Reporting

# Cloud 종류

- Cloud Computing **모델-FaaS**
  - FaaS의 진화 과정

## Evolution of Functions as a Service

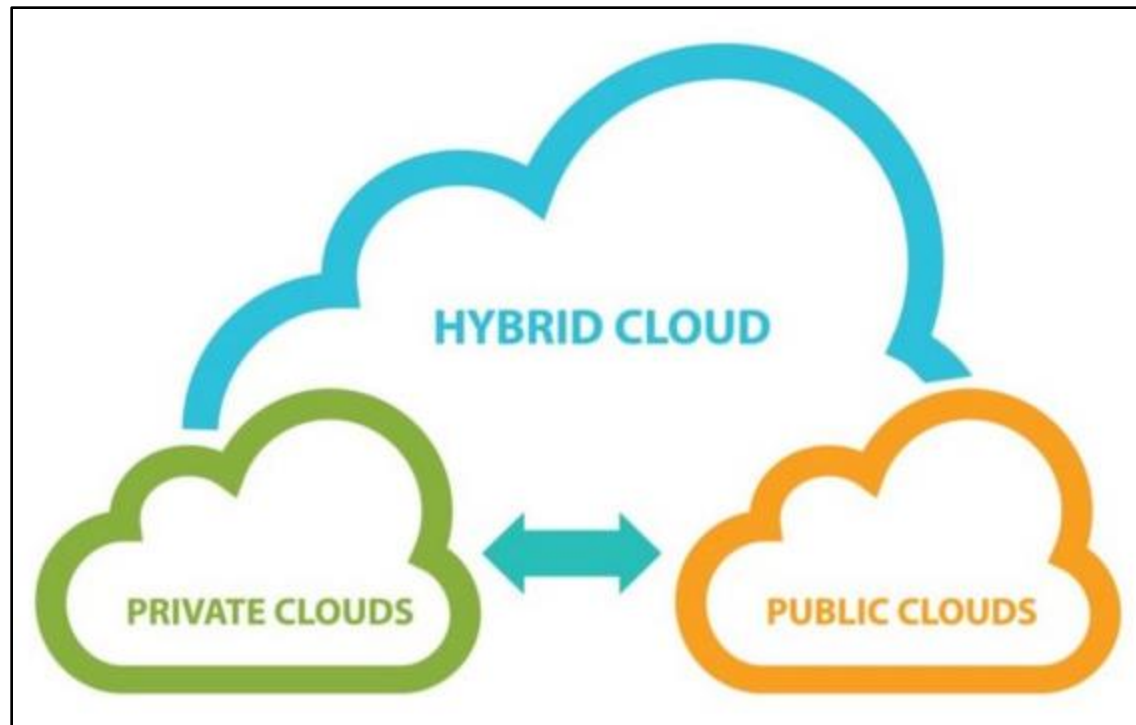




# Cloud 종류

- Cloud Computing 배포

- 어떤 방법으로 **Cloud** 서비스를 제공하느냐에 따라 구분하는 것이다
- Private Cloud
- Public Cloud
- Hybrid Cloud



- Cloud Computing **배포**

- Private Cloud

- 높은 수준의 Customization 가능
    - 초기 비용이 매우 많이 들어간다
    - 유지 보수 비용도 비싸다
    - 높은 보안을 유지할 수 있다

- Public Cloud

- Cloud의 대표로서 일반인들도 이용할 수 있다
    - 낮은 비용과 높은 확장성을 제공한다

- Hybrid Cloud

- Private + Public
    - Private에서 Public으로 전환할 때 이용할 수 있고, 또는 보안 및 편리성을 위해 계속 이 배포 모델을 유지할 수 있다

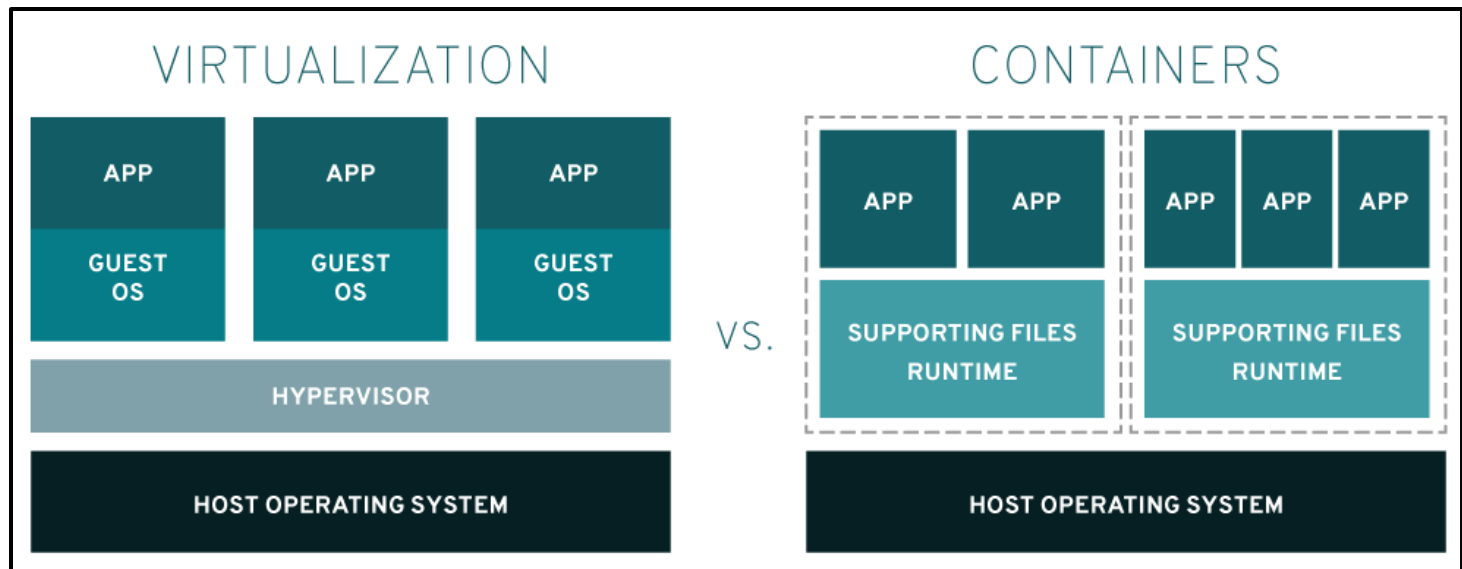
## 2-Virtualization vs Container



# Virtualization vs Container

- Virtualization vs Container

- 서버 서비스를 물리적 컴퓨터보다는 가상 컴퓨터 및 Container로 하는 것이 유리하다
- Virtualization
  - 서버 가상화, 데스크톱 가상화(VDI), 네트워크 가상화, 스토리지 가상화, 응용프로그램 가상화, Presentation 가상화
- Container: docker, Kubernetes



# Virtualization

- Virtualization 장점

- 물리적 서버의 한계는 **유연성의 부족**과 **서버 자원을 낭비**하는 것이다
  - 서비스의 안정성과 보안을 위해서 하나의 서버에 하나의 서비스만 운영
- 가상 서버는 **남아도는 자원을 활용하기** 위해서 만들어졌다
  - 한 대의 물리적인 서버에 Hypervisor를 설치하고 RAM, CPU도 추가하여 10대 이상의 VM을 운영할 수 있다

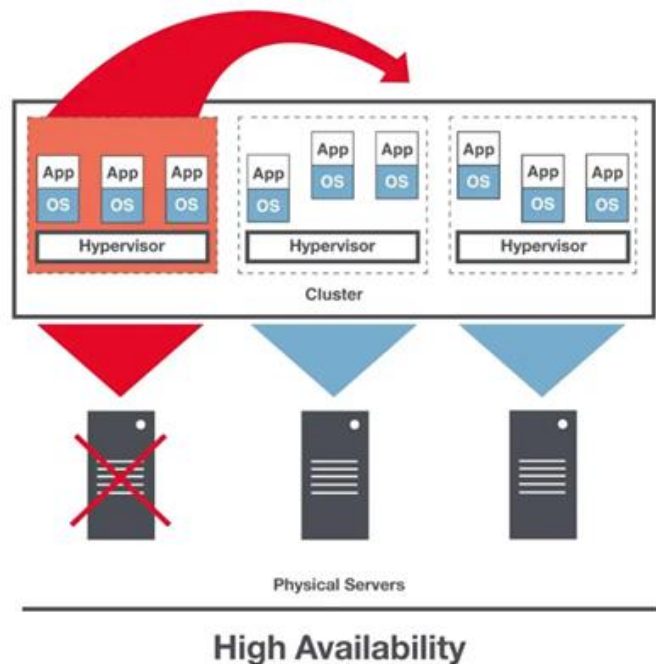


- 노후된 물리적 서버를 **가상서버로 Migration**하고, 가용성을 높이기 위해 Clustering을 구축하여 **VM을 Live Migration**도 할 수 있다

# Virtualization

- Virtualization 장점

- Clustering을 지원하지 못하는 Service도 VM에 해당 서비스를 운영하여 VM을 클러스터링 하면 해당 서비스를 Clustering을 할 수 있다
- 가상화로 서비스를 할 때는 Clustering이 필수적인 조건이다
  - Hyper-V Host가 고장 나면 10대의 VM의 서비스가 중단되기 때문이다



- 현재 Cloud Service는 모두 VM으로 하고 있다


# Virtualization

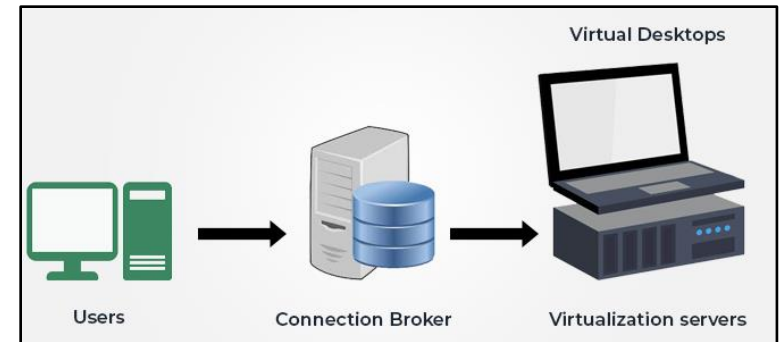
- 가상화 환경 vs 물리적 환경

가상화 환경		물리적 환경	
장점	단점	장점	단점
H/W 비용 및 설치 공간 절감	초기 비용 발생	초기 비용 발생	장비 노후화 시 교체 파트 비용 발생
가동시간 및 가용성 향상			장비 교체 또는 추가 시 새로운 H/W 비용 발생
보안 및 데이터 보호 향상			관리 대상 컴퓨터 증가
서버 및 응용 프로그램 관리 간소화			이중화 시 별도 장비 구성 필요
재해 복구 및 비즈니스 연속성 구현			

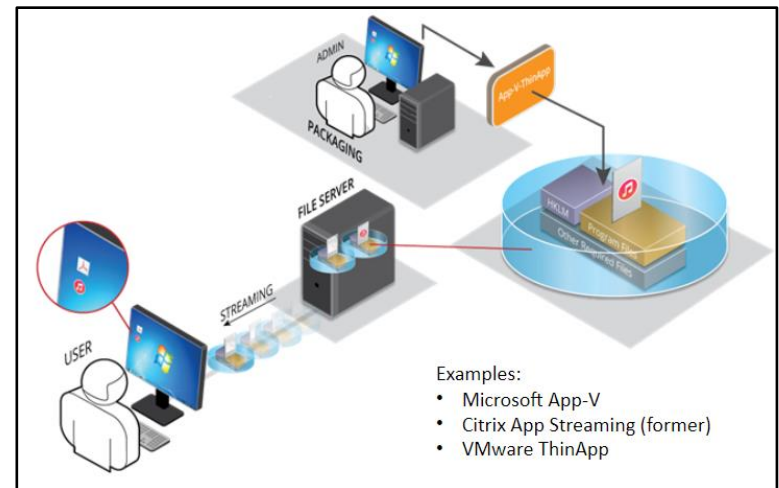
# Virtualization

- Virtualization 종류

- Server Virtualization: 가상화의 가장 기본이고 핵심
  - Desktop Virtualization(VDI; Active Directory 필수)
    - BYOD
    - 보안 및 업무 효율화
    - 네트워크 성능에 영향을 받음
- 



- Network Virtualization
- Storage Virtualization
- Application Virtualization



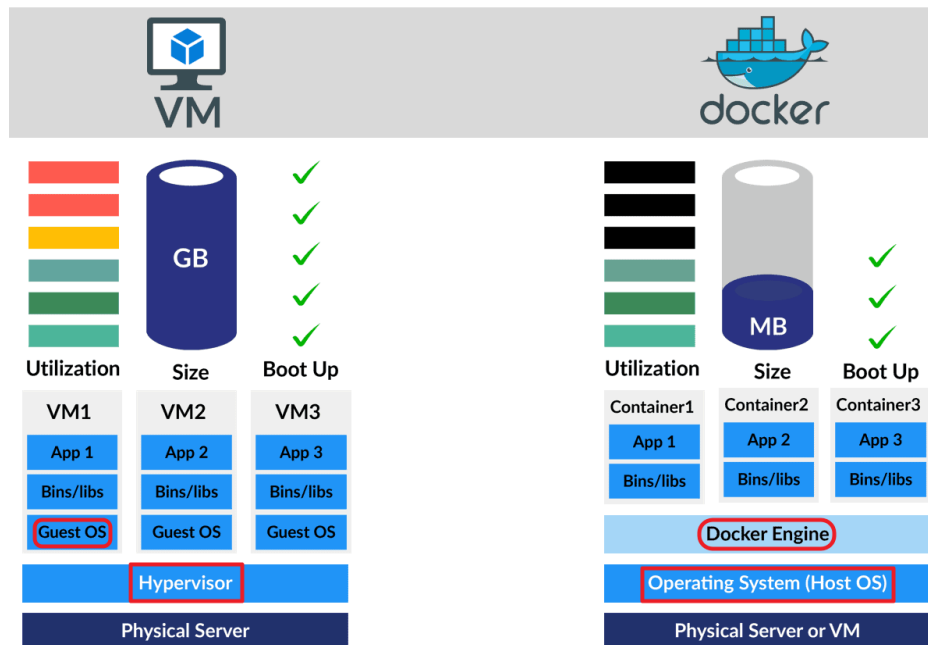
- Presentation Virtualization



# Container

- Container(Kubernetes) 장점

- VM에 비하여 Application **용량 작고 배포 시간 짧음**
- Host의 Kernel을 Container가 공유하기 때문에 **VM에 비해 성능 우수**
- 부하 발생 시 신속한 Application의 **Scale out**
- 갑작스런 서비스 중단 및 서비스 불안정 시 자동으로 서비스 재시작(초기화)하여 정상화 구현(**관리자 개입 불필요**)
- 단계적인 서비스 업그레이드 및 서비스 불안정시 이전 버전으로 롤백



# Container

- Docker 사용 예

- 여러 **OS** 동시 사용하기

- `docker run --name myubuntu -itd ubuntu`
    - `docker exec -it myubuntu /bin/bash`
    - `docker run --name myalpine -itd alpine`
    - `docker exec -it myalpine /bin/sh`

- 여러 개의 동일한 **포트**를 사용하여 Web Application 동시 실행하기

- `docker run --name myweb1 -d nginx`
    - `docker run --name myweb2 -d nginx`
    - `docker ps`
    - `docker exec -it myweb1 /bin/bash`
    - `curl http://localhost`

- **Game** 실행하기

- `docker run --name mygame -d -p 80:8080 pengbai/docker-supermario`



# Container

- Docker 사용 예

- Container 관리 프로그램 운영(**GUI로 운영**)

- **curl http://down.cloudshell.kr/docker/portainer.sh | bash**
    - http://ip:9000

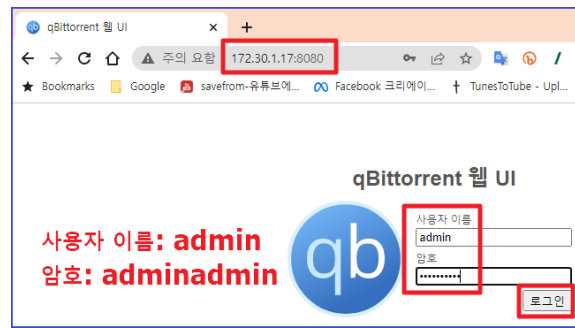
- **Database** 사용하기

- docker run --name mysqlcon -d -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=password mysql
    - docker exec -it mysqlcon bash
    - mysql -u root -p

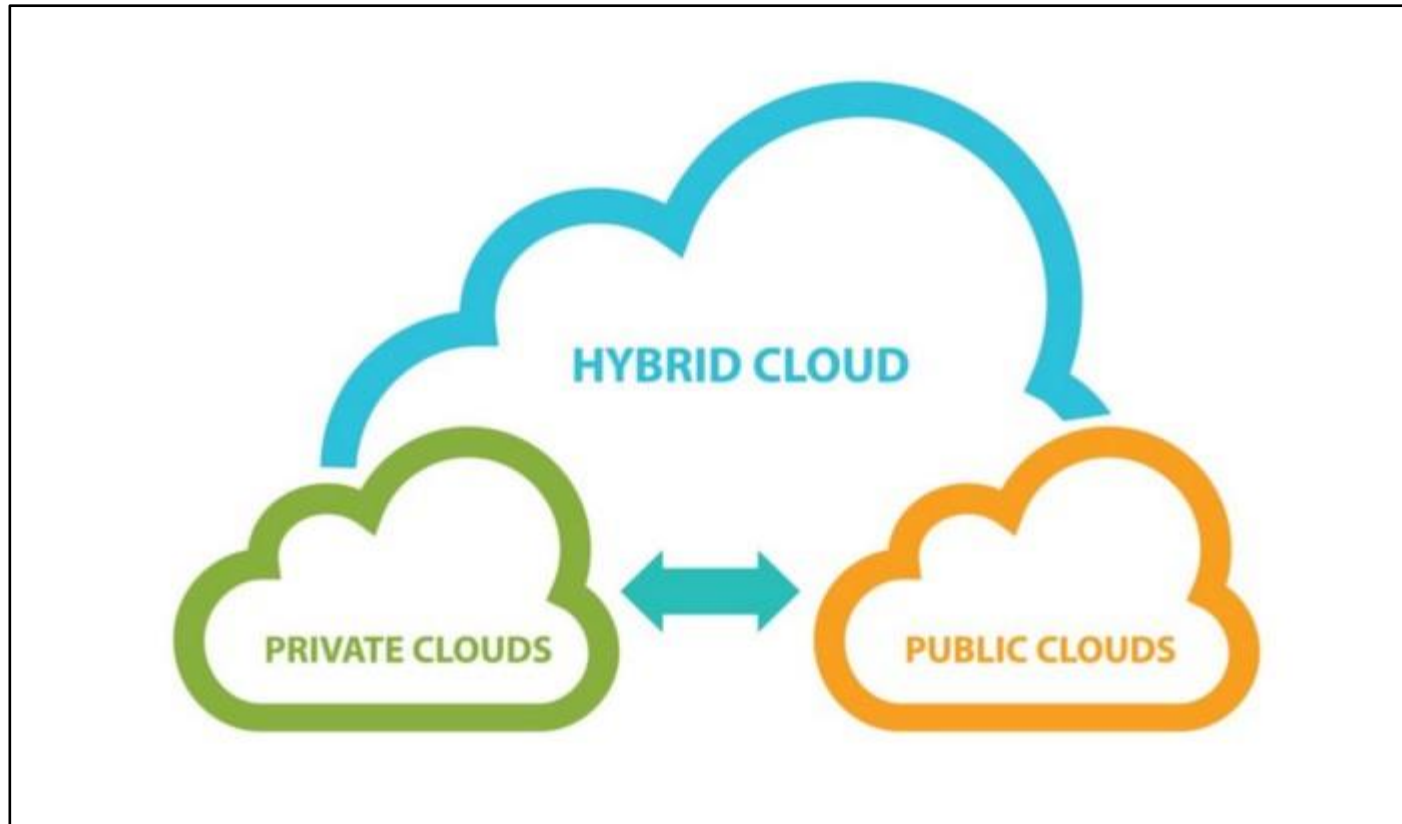
- qbittorrent 컨테이너 실행하기

- **curl http://down.cloudshell.kr/docker/qbittorrent.sh | bash**

- Chrome을 실행하여 8080 포트로 접속하여 토렌트 파일 경로를 입력하여 사용한다

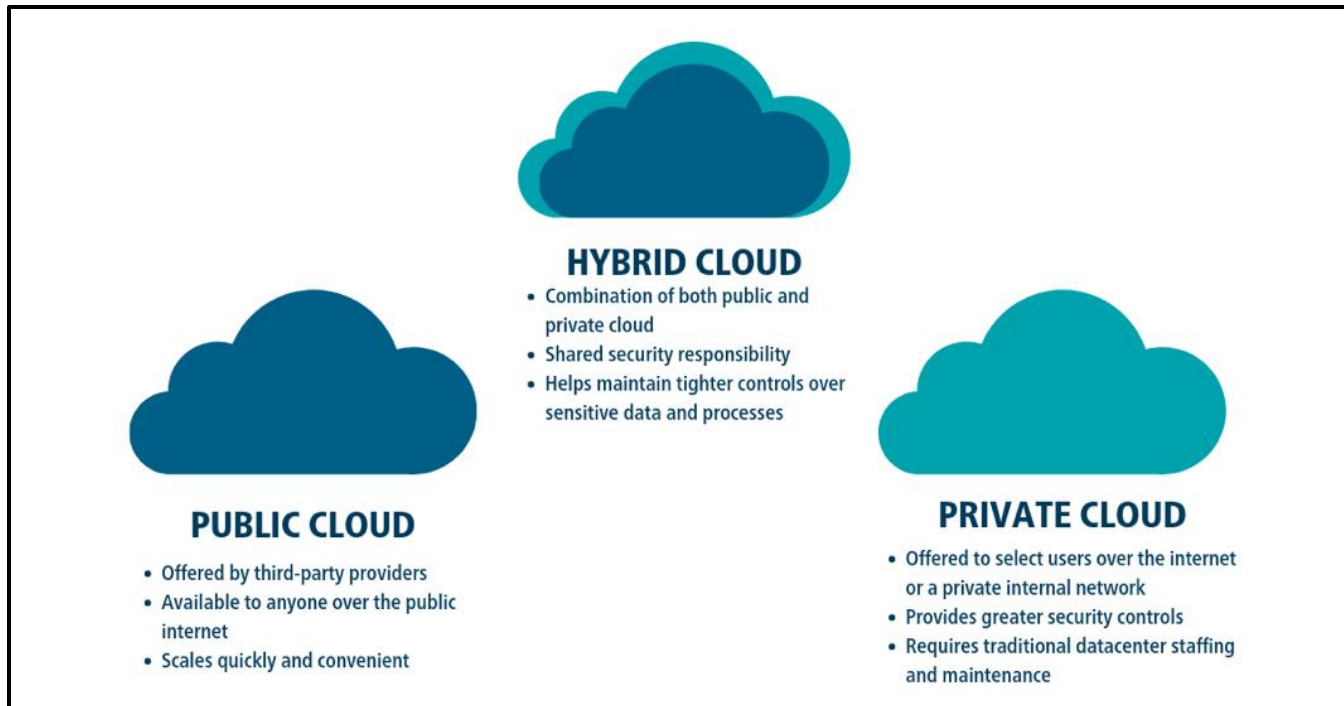


### 3-Private Cloud vs Public Cloud



# Private Cloud vs Public Cloud

- Private Cloud vs Public Cloud
  - 가상화에서 발전한 것이 Private Cloud(사용자 스스로 VM 생성)
  - Private Cloud는 데이터 보호
    - Open Stack, Azure Stack
  - Public Cloud가 대세, 사용한 만큼 과금
    - AWS, Azure, GCP



# Private Cloud

- Private Cloud란?

- 기업이 직접 클라우드 환경을 구축하고 이를 **기업 내부에서 활용하거나 또는 계열사에 공개하는 것**을 말한다
- 외부 클라우드 사업자의 서비스를 이용하지 않고 서비스를 위한 인프라를 **직접 구축**한다는 점에서 프라이빗 클라우드는 자체 인프라 구축과 동일하다
- Private Cloud와 On-Premises의 가장 큰 차이점은 '**신속함**'이다
  - 기업의 서비스에 사용자가 몰리면 **Private Cloud는 해당 서비스를 위한 인프라를 증설해** 빠르게 서비스를 안정화시킬 수 있다
  - 반면 On-Premises는 인프라를 새로 주문하고 이를 설치해 서비스에 연결할 때까지(보통 1~2주) 서비스를 안정시킬 수 없다
- Private Cloud가 신속함을 갖추기 위해 두 가지 핵심 능력 필요
  - 가상화
  - 클라우드 관리 스택
    - **Azure Stack, Open Stack, IBM Cloud Private, Cloud Stack**

# Public Cloud

- Public Cloud란?

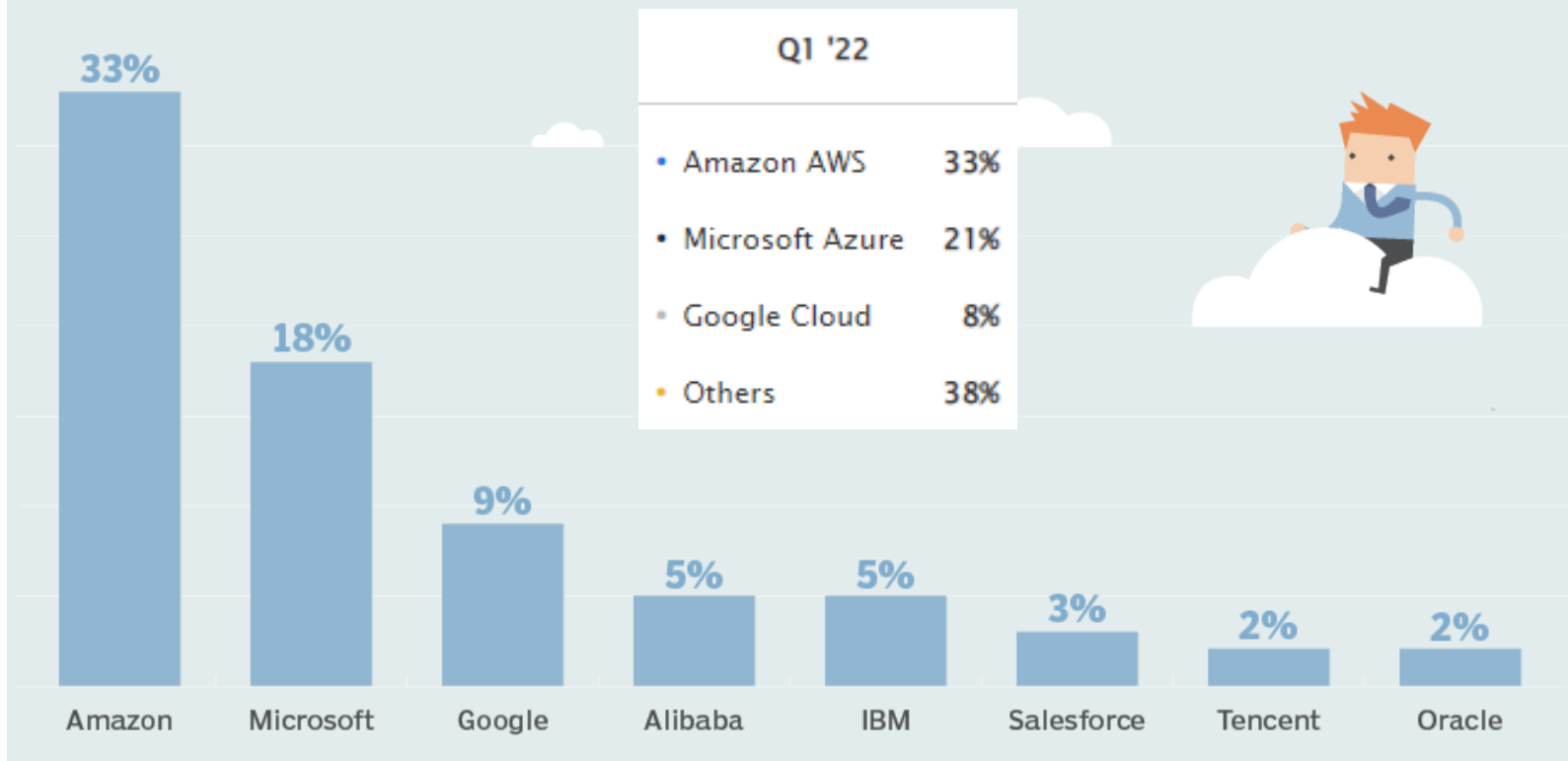
- 외부 클라우드 사업자가 제공하는 서비스를 통해 클라우드를 이용하는 것으로, 우리가 가장 흔히 접할 수 있는 서비스 형태다
- 서비스를 위한 **모든 인프라를 클라우드 업체에서 제공받는다. 자체 인프라가 없거나 빈약한 스타트업, 중견기업이 많이 이용하고 있다**
  - Public Cloud가 주도권 및 대세
- Public Cloud는 4차 산업혁명 시대의 핵심 기술이고, 21세기의 은행이며, 동시에 황금알을 낳는 거위다
- 주요 Public Cloud Provider
  - AWS(Amazon Web Service)
  - Azure
  - GCP(Google Cloud Platform)
  - Alibaba Cloud(오픈 스택 + 알리바바 자체 기술)
  - IBM Cloud(이전 이름: IBM Bluemix and IBM SoftLayer)
  - 네이버 비즈니스 플랫폼(자체 기술 기반 클라우드 관리 스택)

# Public Cloud

- Public Cloud 시장 점유율

## Worldwide public cloud market share

Here are the top IaaS, PaaS and hosted private cloud providers





# Public Cloud

- Public Cloud가 제공하는 각종 서비스 비교
  - 참고: <https://bit.ly/3EDB8Zz>

용도	Google Cloud Platform	AWS	Azure
CI/CD	Cloud Build	AWS CodeBuild, AWS CodeDeploy, AWS CodePipeline	Azure DevOps, Github Enterprise
Execution Control	Cloud Tasks	Amazon Simple Queue Service (SQS) Amazon Simple Notification Service (SNS)	Azure Service Bus Azure Storage Queues
Multi-cloud	Anthos	Amazon EKS Anywhere Amazon ECS Anywhere	Azure Arc
Service mesh	Anthos Service Mesh	AWS App Mesh	Azure Service Fabric
Service mesh	Istio on Google Kubernetes Engine	Istio on Amazon EKS	Istio in Azure Kubernetes Service
ML	Cloud GPU	Amazon Elastic Compute Cloud(EC2) P3	GPU Optimized VMs
ML	Cloud TPU	AWS UltraClusters	Azure Virtual Machines
VM	Compute Engine	Amazon Elastic Compute Cloud(EC2)	Azure Virtual Machines
AutoScaling	Compute Engine Autoscaler	AWS Autoscaling	Azure Autoscale, Azure Virtual Machine Scale Sets

Block Storage	Persistent Disk	Amazon Elastic Block Store (EBS)	Azure Managed Disks
serverless function	Cloud Functions	AWS Lambda	Azure Functions Serverless Compute
fully managed serverless platform	App Engine	AWS Elastic Beanstalk	Azure App Service
fully managed serverless platform	Cloud Run	AWS Fargate, AWS Lambda, AWS App Runner	Azure Container Instances
Kubernetes	Google Kubernetes Engine	Amazon Elastic Kubernetes Service (EKS) Amazon Elastic Container Service (ECS)	Azure Kubernetes Service (AKS)
Container registry	Artifact Registry	Amazon Elastic Container Registry (ECR)	Azure Container Registry
Data Integration / ETL	Cloud Data Fusion	Amazon AppFlow, Amazon Data Pipeline, AWS Glue	Azure Data Factory
Data warehouse	Big Query	Amazon Athena, Amazon Redshift	Azure Synapse Analytics

# Cloud 종류

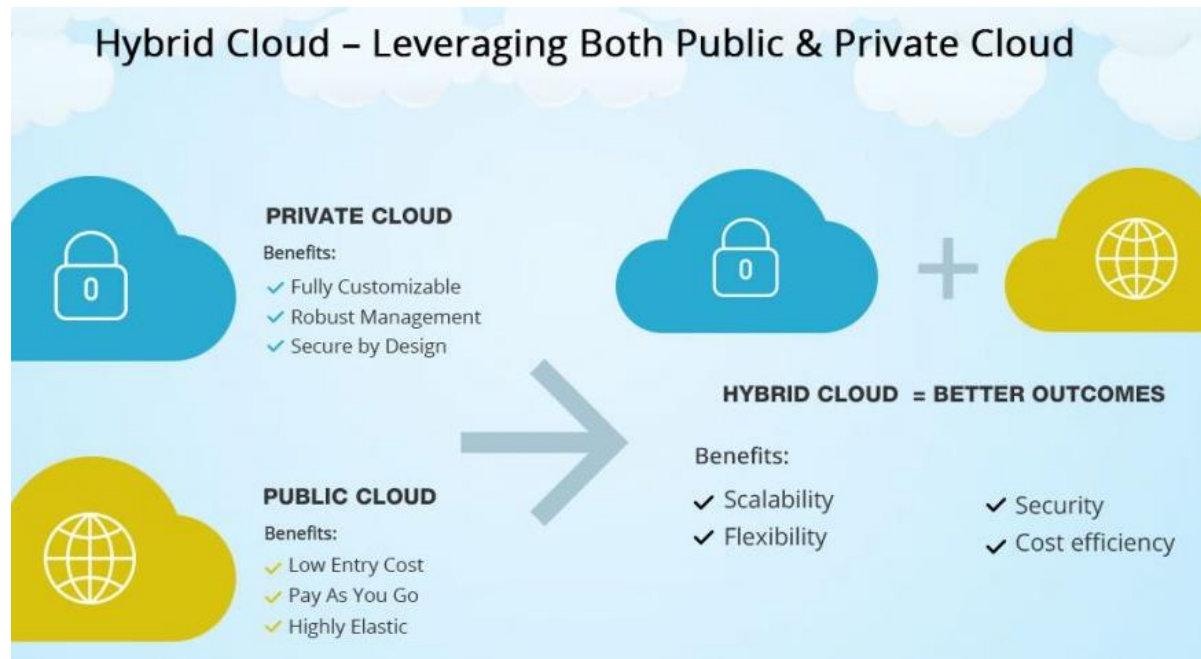
- Public Cloud가 제공하는 각종 서비스 이름 비교표

	 AWS	 Azure	 Google Cloud
Available Regions	AWS Regions and Zones	Azure Regions	Google Compute Regions & Zones
Compute Services	 Elastic Compute Cloud (EC2)	 Virtual Machines	 Compute Engine
App Hosting	 Amazon Elastic Beanstalk	 Azure Cloud Services	 Google App Engine
Serverless Computing	 AWS Lambda	 Azure Functions	 Google Cloud Functions
Container Support	 Elastic Container Service	 Azure Container Service	 Container Engine
Scaling Options	 Auto Scaling	 Azure Autoscale	 Autoscaler
Object Storage	 Amazon Simple Storage (S3)	 Azure Blob Storage	 Cloud Storage
Block Storage	 Amazon Elastic Block Storage	 Azure Managed Storage	 Persistent Disk
Content Delivery Network (CDN)	 Amazon CloudFront	 Azure CDN	 Cloud CDN
SQL Database Options	 Amazon RDS	 Azure SQL Database	 Cloud SQL
NoSQL Database Options	 AWS DynamoDB	 Azure DocumentDB	 Cloud Datastore
Virtual Network	 Amazon VPC	 Azure Virtual Network	 Cloud Virtual Network
Private Connectivity	 AWS Direct Connect	 Azure Express Route	 Cloud Interconnect
DNS Service	 Amazon Route 53	 Azure Traffic Manager	 Cloud DNS
Log Monitoring	 Amazon CloudTrail	 Azure Operational Insights	 Cloud Logging
Performance Monitoring	 Amazon CloudWatch	 Azure Application Insights	 Stackdriver Monitoring
Administration and Security	 AWS Identity and Access Management (IAM)	 Azure Active Directory	 Cloud Identity and Access Management (IAM)
Compliance	 AWS CloudHSM	 Azure Trust Center	 Google Cloud Platform Security
Analytics	 Amazon Kinesis	 Azure Stream Analytics	 Cloud Dataflow
Automation	 AWS Opsworks	 Azure Automation	 Compute Engine Management
Management Services & Options	 Amazon CloudInformation	 Azure Resource Manager	 Cloud Deployment Manager
Notifications	 Amazon Simple Notification Service (SNS)	 Azure Notification Hub	None
Load Balancing	 Elastic Load Balancing	 Load Balancing For Azure	 Cloud Load Balancing

# Hybrid Cloud

- Hybrid Cloud란?

- 외부 클라우드 사업자가 제공하는 클라우드 서비스와 자체 인프라(온프레미스)를 함께 활용하는 것이다
- 보통 서비스 구동은 클라우드 상에서, 데이터 보관이나 로컬 서비스는 자체 인프라에서 처리하는 형태로 구현되고 있다
- 클라우드 서비스의 이점을 누리면서 데이터를 직접 관리하고 싶은 대기업들이 주로 활용한다



# Private vs Public vs Hybrid

- Public vs Private vs Hybrid

## ***PUBLIC CLOUD***

A multi-tenanted environment operated by a third party service provider in which businesses pay for provisioned services.

### ***ADVANTAGES***

- No Capital Cost
- Low IT Overheads
- Infinite Scalability

• • • • •

### ***DISADVANTAGES***

- Lack of Customization
- Governance Issues
- Potential Latency

## ***PRIVATE CLOUD***

A single-tenanted environment over which businesses have complete control with regard to architecture and configuration.

### ***ADVANTAGES***

- Fully Customizable
- Higher Level of Security
- Superior Performance

• • • • •

### ***DISADVANTAGES***

- Capital Cost
- Under Utilization
- High IT Overheads

## ***HYBRID CLOUD***

A combination of public and private environments, offering the advantages of both with fewer disadvantages.

### ***ADVANTAGES***

- Greater Flexibility
- Resilience to Outages
- No Capacity Ceiling
- Fewer IT Overheads
- Manageable Security

• • • • •

### ***DISADVANTAGES***

- Compatibility

# Multi-Cloud

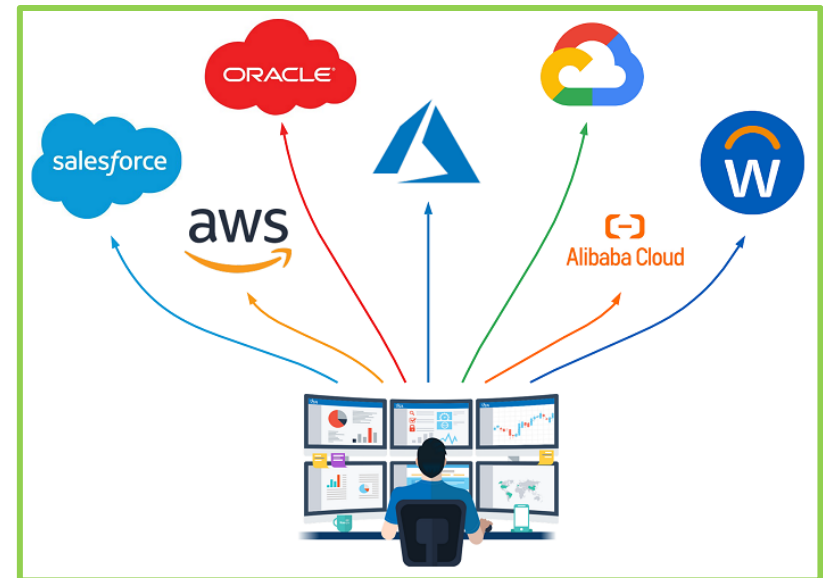
- Multi-Cloud란?

- 두 개 이상의 Cloud Vendor가 제공하는 Public 및 Private Cloud로 구성된 Cloud에 접근하는 방식이다
  - 기업이 **단일 벤더에 의존했을 때 발생할 수 있는 위험을 피할 수 있다**
  - 여러 클라우드 벤더에 워크로드를 분산하면 필요할 때 언제든지 클라우드를 사용하거나 중지할 수 있는 유연성이 확보된다
  - 여러 클라우드를 운용하는 것은 문제가 되지 않는다

Bare metal 서버에서 Virtualization 기반 워크로드로 이전했으며, 사용자 변동이 매우 큰 고객용 App만 중점적으로 지원하기 위해 Public Cloud 옵션을 살펴보고 있다.

해당 조사를 수행한 후, SLA, 보안 프로토콜, Uptime을 적절히 조합하여 Custom App을 호스팅하는 Public Cloud Vendor를 찾아서 만족하고 있다.

그러나 얼마 지나자 고객들은 다른 벤더의 독점적인 App에서만 제공되는 기능을 요구하기 시작한다. 이러한 기능을 Custom App에 통합하기 위해서는 Vendor의 Application을 구매해야 할 뿐 아니라 그 Vendor의 독점적인 Public Cloud에서 Application을 호스팅해야 한다는 상황에 처하게 된다.





# Multi-Cloud

- Single Cloud 사용의 문제점
  - 해당 클라우드에 의존하게 된다(Lock-in)
  - 이슈 발생에 대한 자체 대응이 불가하다
  - 조직별 다양한 특성이 있다
- Multi-Cloud 사용의 필요성
  - Cloud 차원의 고가용성(HA) 구성
  - 다양한 대안 수립
  - 조직 특성 보장
  - Shadow IT 예방
    - Shadow IT 예
      - 회사 업무를 위한 개인 이메일(Gmail) 계정 사용
      - 승인되지 않은 BYOD 사용
      - 회사 IT 팀에서 관리하지 않는 SaaS 서비스 배포

- Multi-Cloud와 Container

- Linux® Container는 고객이 Public Cloud Vendor를 유연하게 선택할 수 있도록 한다
  - 고객은 Custom Application에 대한 Image를 가지고 있기 때문에, 해당 이미지를 원하는 Cloud 업체의 Container 서비스에 배포하여 운영하면 된다
  - 이것은 특정한 App 배포 및 운영에 대한 의존성을 해결할 수 있다
- Container가 전체 Run Time 환경을 통해 애플리케이션을 패키징하고 격리하면 사용자는 이렇게 패키징 방식으로 포함된 애플리케이션을 Cloud들 사이에서 이동하면서 모든 기능을 그대로 사용할 수 있다.
- 따라서 기업이 독점 솔루션의 제약으로 인한 Workload 지원 여부가 아니라 Uptime, Storage 공간, 비용 등의 범용 표준을 기준으로 하여 Public Cloud 제공업체를 선택할 수 있게 된다

# Multi-Cloud

- Application별로 Multi-Cloud에 배포하기
  - 각각의 Public Cloud의 서비스의 특징을 잘 알면 Application 단위로 다른 Cloud를 이용할 수도 있다
    - One for program logic, the second for databases, and the third for machine learning
  - Multi-Cloud를 사용하여 유연성과 다양한 Cloud의 장점 활용 가능

