

## 쿠버네티스를 활용한 멀티클라우드 도입과 운영전략 - AWS, Azure, GCP 비교와 실습

2020년 3월  
아인인텔리전스  
권재원, Ph.D.

# 강좌 소개

- 제목: 쿠버네티스를 활용한 멀티 클라우드 도입과 운영 - AWS, Azure, GCP 비교와 활용 실습
- 목표: 쿠버네티스를 활용하여 멀티클라우드를 도입하고 전략적으로 운영하는 방법을 학습합니다.
- 대상:
  - 클라우드를 사용하고 있거나 멀티 클라우드 도입을 검토하는 분
  - Agile DevOps 체계 도입에 관심이 많거나 도입을 검토하시는 분
- 나이도: 중급 (Series 200~300 수준)
- 사전 보유 지식
  - 기초 리눅스 명령어
  - 인터넷 네트워크 시스템에 대한 기본 지식
  - 가상화에 대한 기초적 지식
  - 기본적 인터넷 보안 및 계정관리에 관한 지식
  - 클라우드 사용 경험
- AWS, Azure, GCP 를 모두 실습함

## 교육 순서

1일차

Module 1 : Multi-Cloud 이해와 AWS vs. Azure vs. GCP 비교

Module 2 : 가상화 / 도커 컨테이너 / 쿠버네티스 핵심개념과 실습

2일차

Module 3 : GCP GKE를 이용한 Kubernetes 배포운영 이해와 실습

3일차

Module 4 : Amazon EKS를 이용한 Kubernetes 배포운영 이해와 실습

Module 5 : MS AKS를 이용한 Kubernetes 배포운영 이해와 실습

4일차

Module 6 : Multi-Cloud에서의 Agile DevOps와 MSA 이해와 환경 구성

Module 7 : Jenkins를 이용한 CI & CD 이해와 실습

5일차

Module 8 : Multi-Cloud에서의 IAM / Security / Compliance 관리

Module 9 : Multi-Cloud Migration & Cost & 운영 전략

Module 10 : Multi-Cloud Architecture 설계 원리 및 사례

**Multi-Cloud & Kubernetes**

쿠버네티스를 활용한 멀티클라우드 도입과 운영전략

## **Module 1**

# **Multi-Cloud 이해와 AWS vs. Azure vs. GCP 비교**

## **Part I**

# **4차 산업혁명과 클라우드 컴퓨팅**

어떤 내용의 사진일지 맞춰보세요



## \* 비교

2005년 1월초 세계 최대 App Server CyWorld  
~ 1200만 가입자 & 700 만 download in 4 days



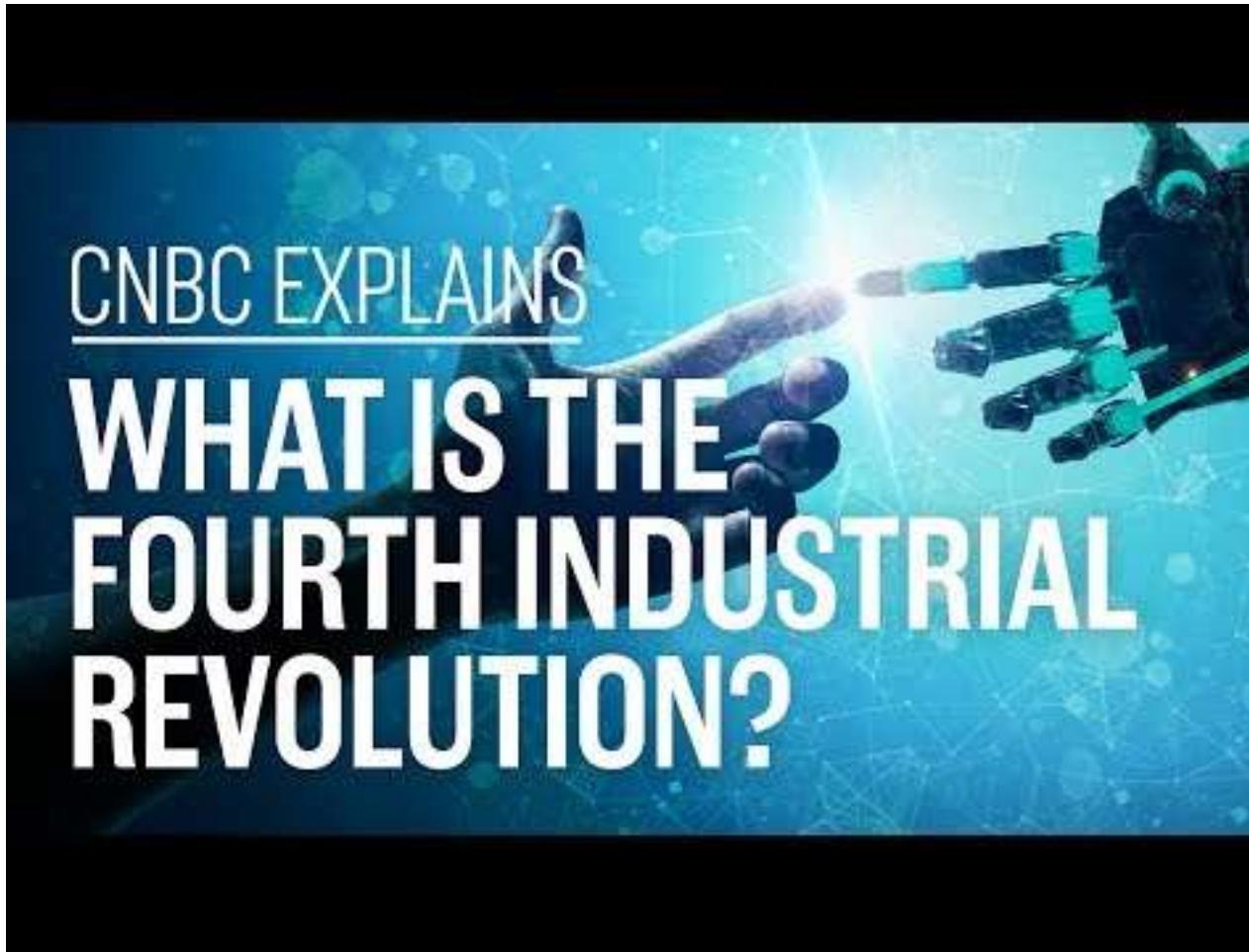
### 온라인 리스크, 어떻게 대응할 것인가

문권모 | 2005-02-18

연예인 X-파일의 사례를 보면 실제로 인터넷 컨텐츠가 확산되는 속도를 가늠해 볼 수 있다. 최근 행해진 인터넷 조사를 종합해 보면 네티즌 중 절반, 즉 국민 대다수가 X-파일을 받아보는 데 걸린 시간은 불과 4일도 되지 않는 것으로 나타났다(뒤페이지 박스기사에 있는 Kryptonite 사례의 경우 단지 열흘 동안에 관련 사실을 열람한 네티즌이 1,800만 명에 이르렀다).

## 4차 산업혁명과 데이터 처리

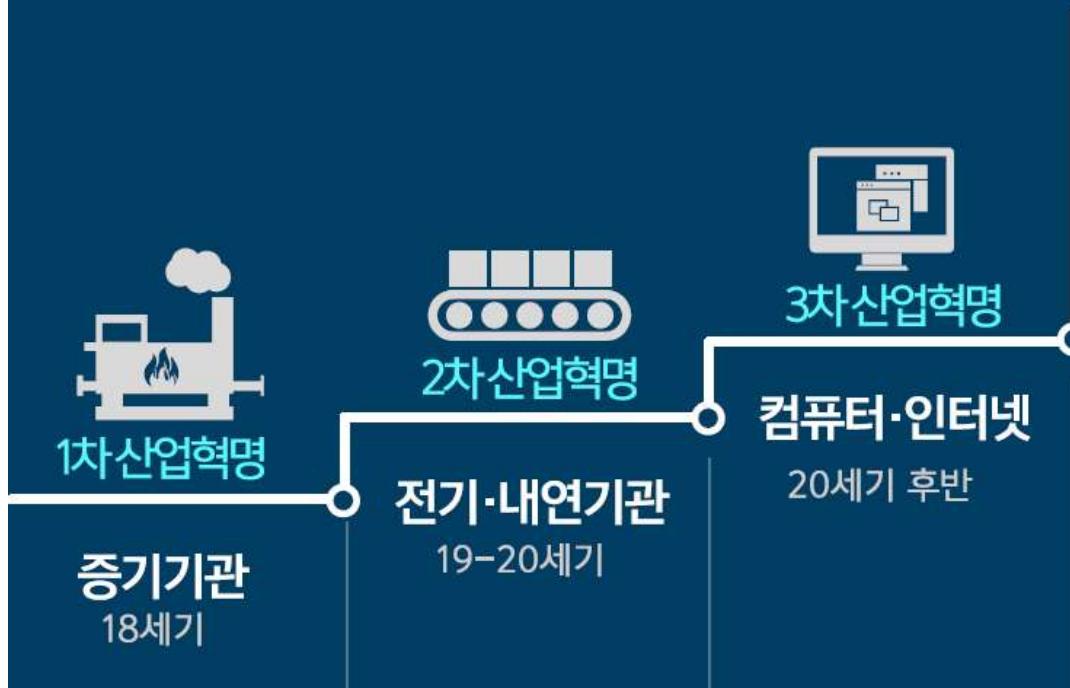
## 4차 산업혁명이란 ?



# 산업의 진화

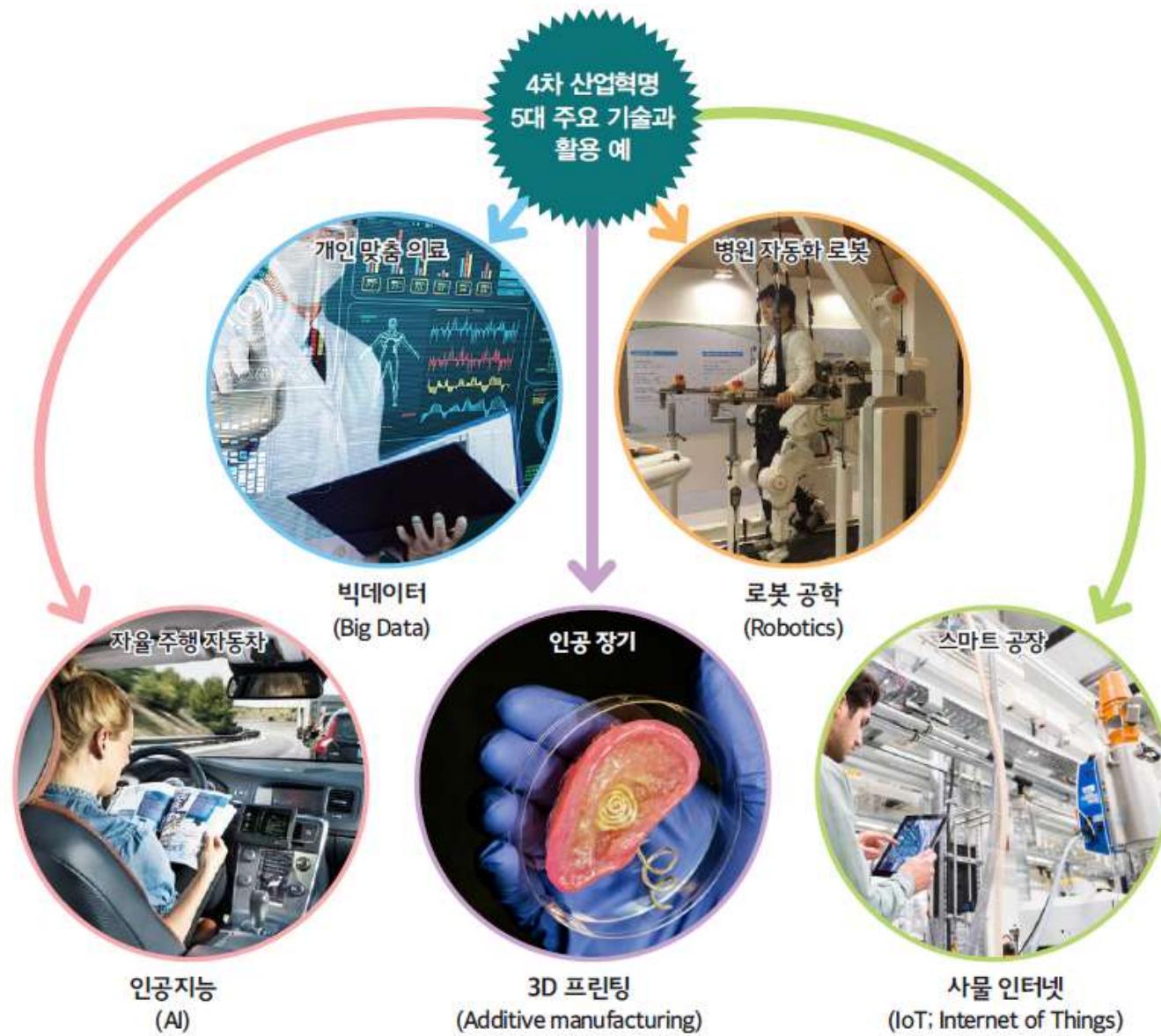
4차 산업혁명이란 무엇인가?

## 파괴적 기술과 역사적 산업혁명의 전개



IoT  
로봇  
AI  
3D  
프린팅  
AR/VR

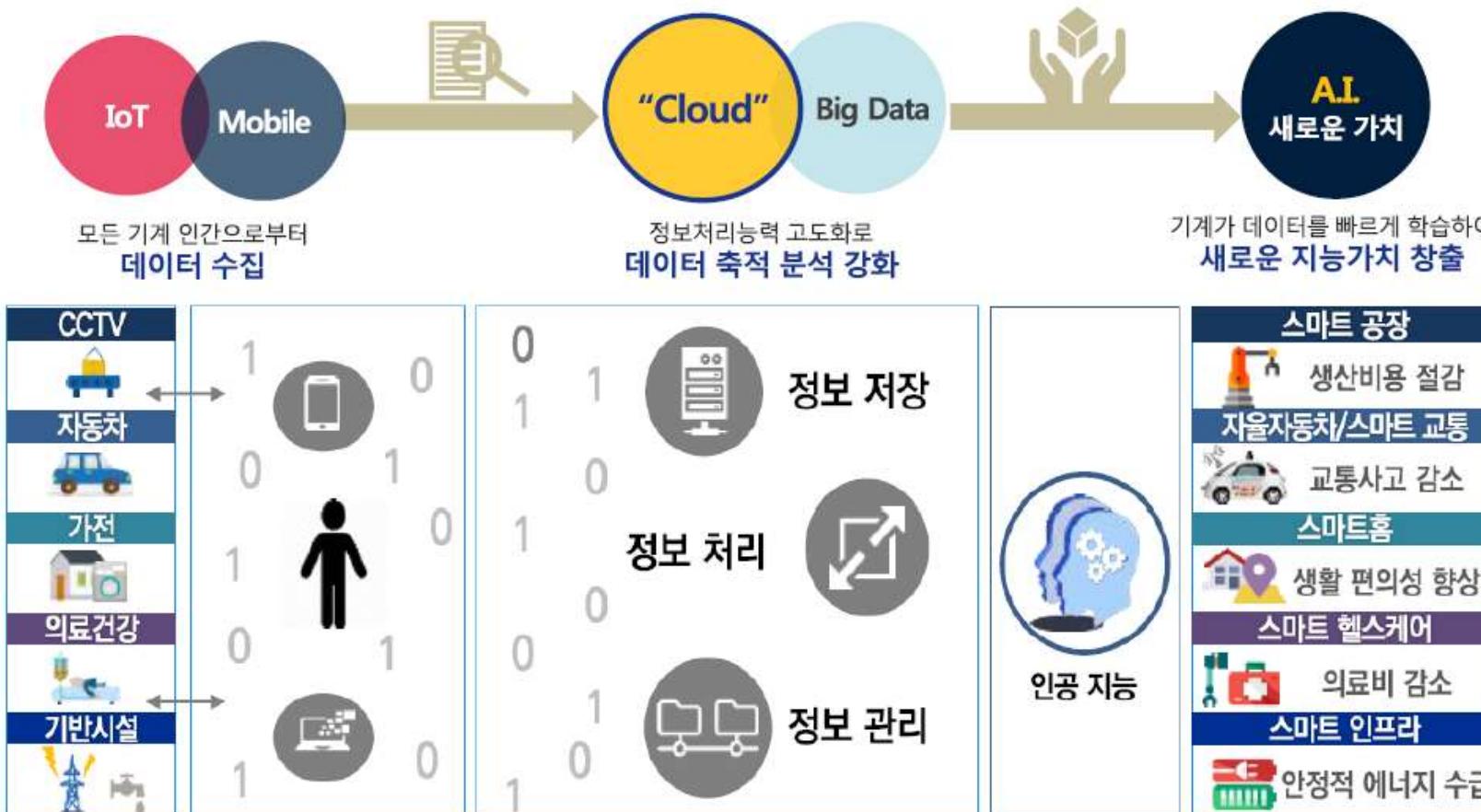
# 4차 산업혁명의 5대 기술



# 4차 산업혁명에서 클라우드 컴퓨팅의 역할

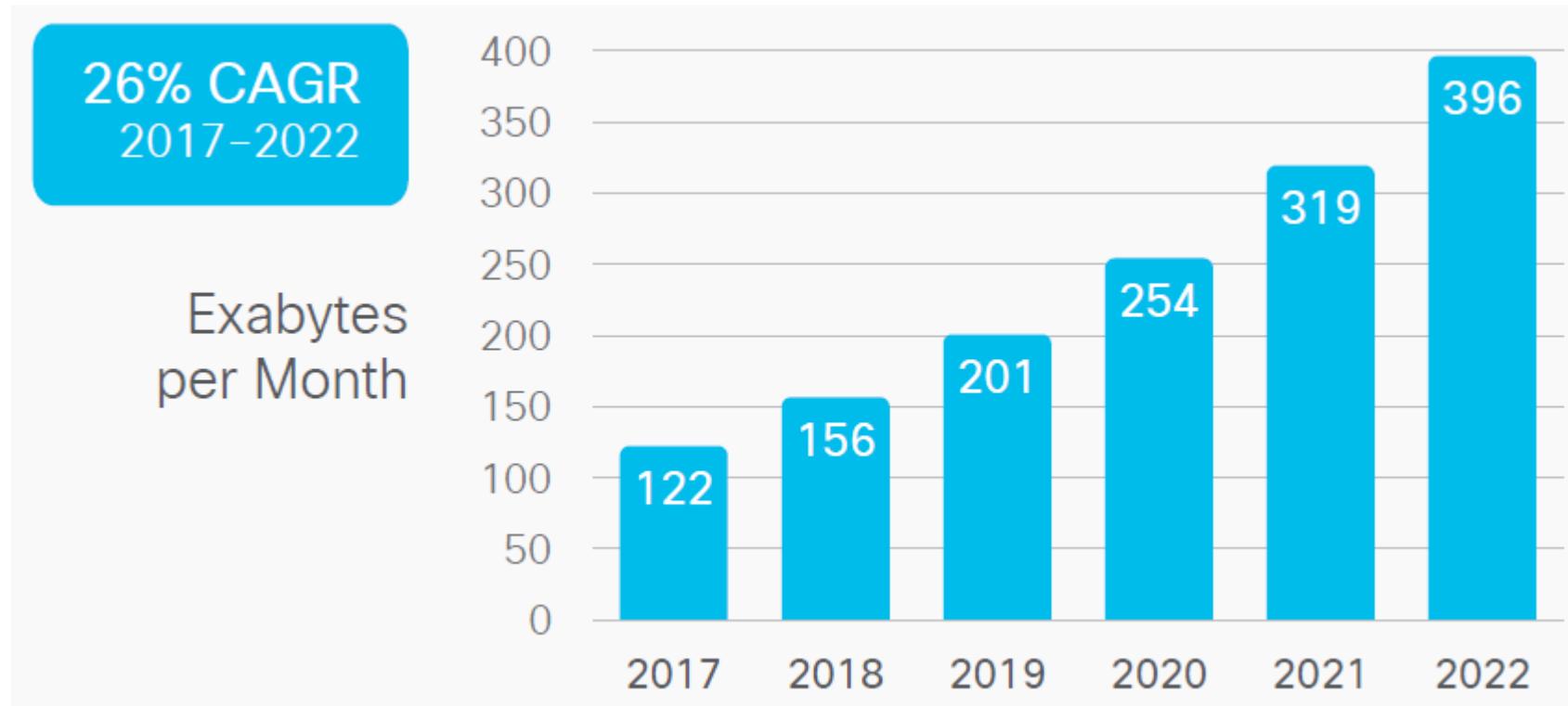
- 4차 산업혁명의 5대 기술인 인공지능/빅데이터, IoT, 로봇, 가상현실, 3D 프린팅을 실현하기 위해서는,
- 기본 Infra인 초고속 통신망과 클라우드 컴퓨팅이 필요

지능정보기술 : AI + ICBM 기술의 획기적 발전 및 폭넓은 연결 확장



## 데이터 트래픽의 증가

- 시스코(Cisco)에 따르면, Visual Networking Index(VNI)는 2022년까지 IP traffic 396 EB per month로 증가한다고 예상 (\* EB = Exabyte = 엑사바이트 =  $10^{18}$  byte)

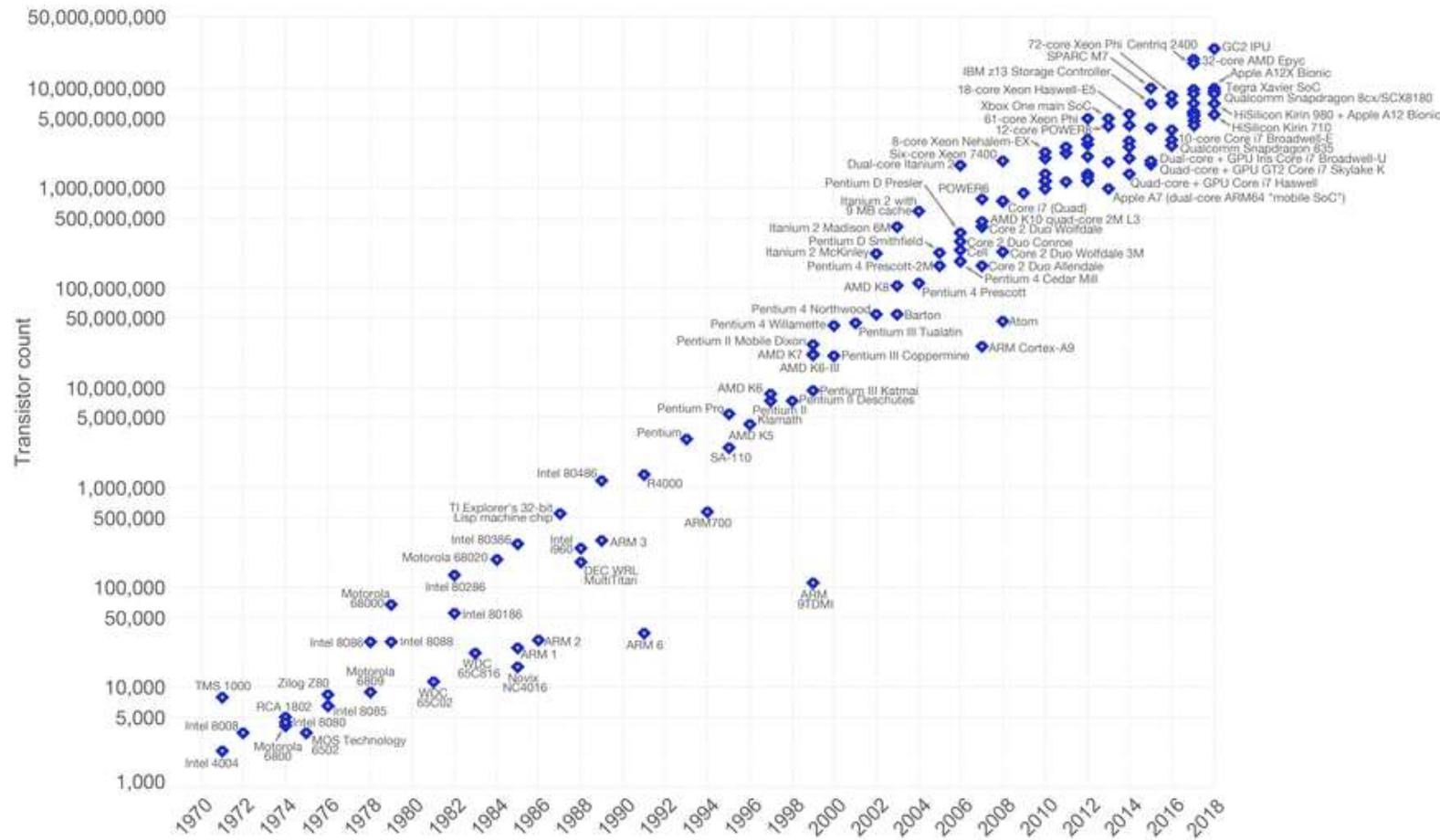


Cisco (2019), 'Cisco VNI Global IP Traffic Forecast, 2017–2022'

# 데이터 트래픽 증가요인 : 연산능력의 지속적 향상 (무어의 법칙)

## Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.

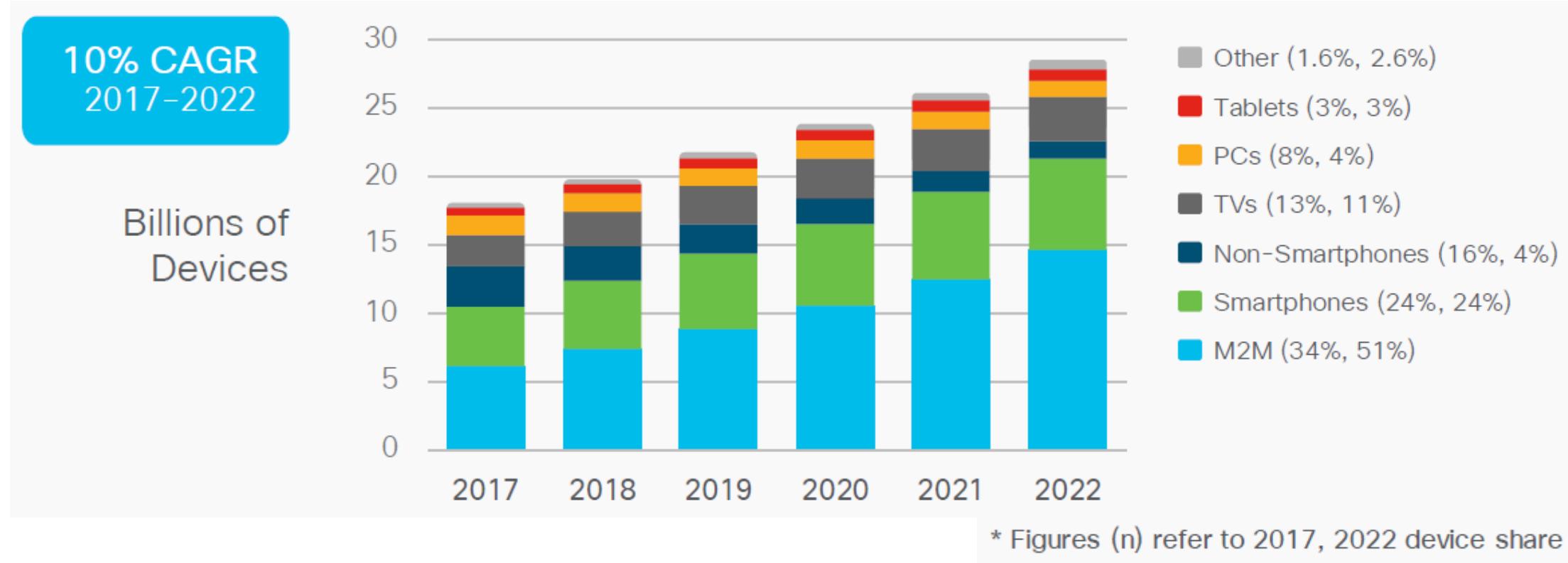


Data source: Wikipedia ([https://en.wikipedia.org/wiki/Transistor\\_count](https://en.wikipedia.org/wiki/Transistor_count))

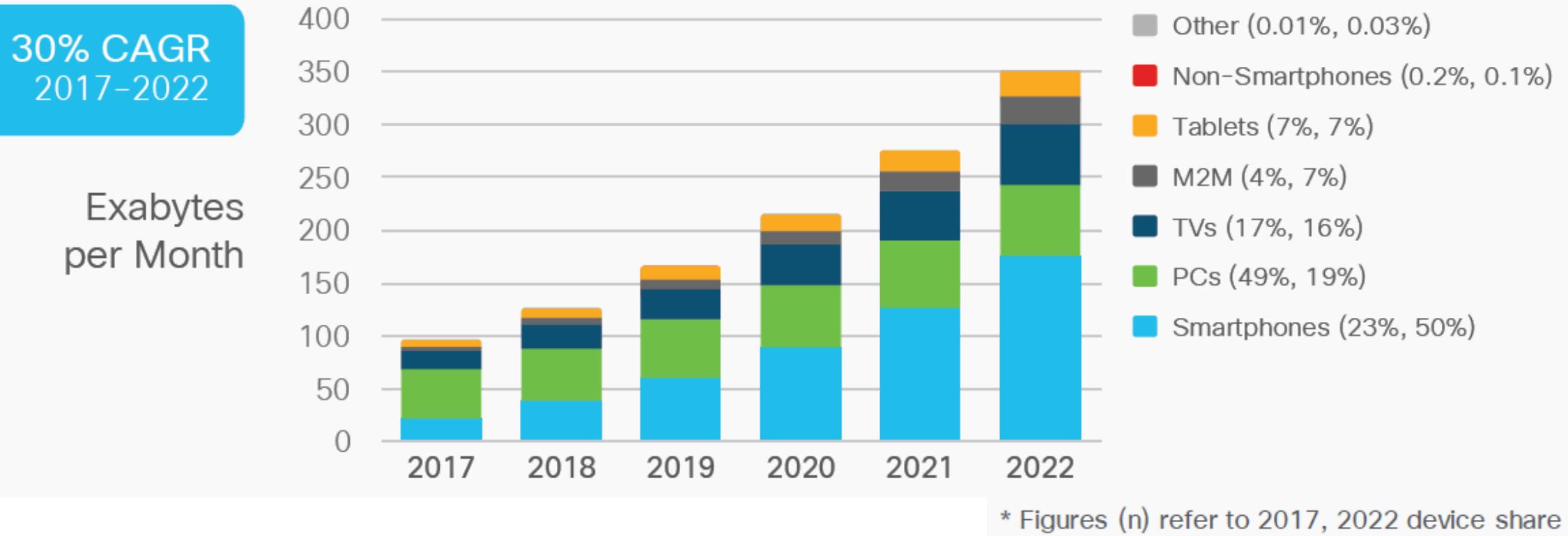
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Licensed under CC-BY-SA by the author Max Roser.

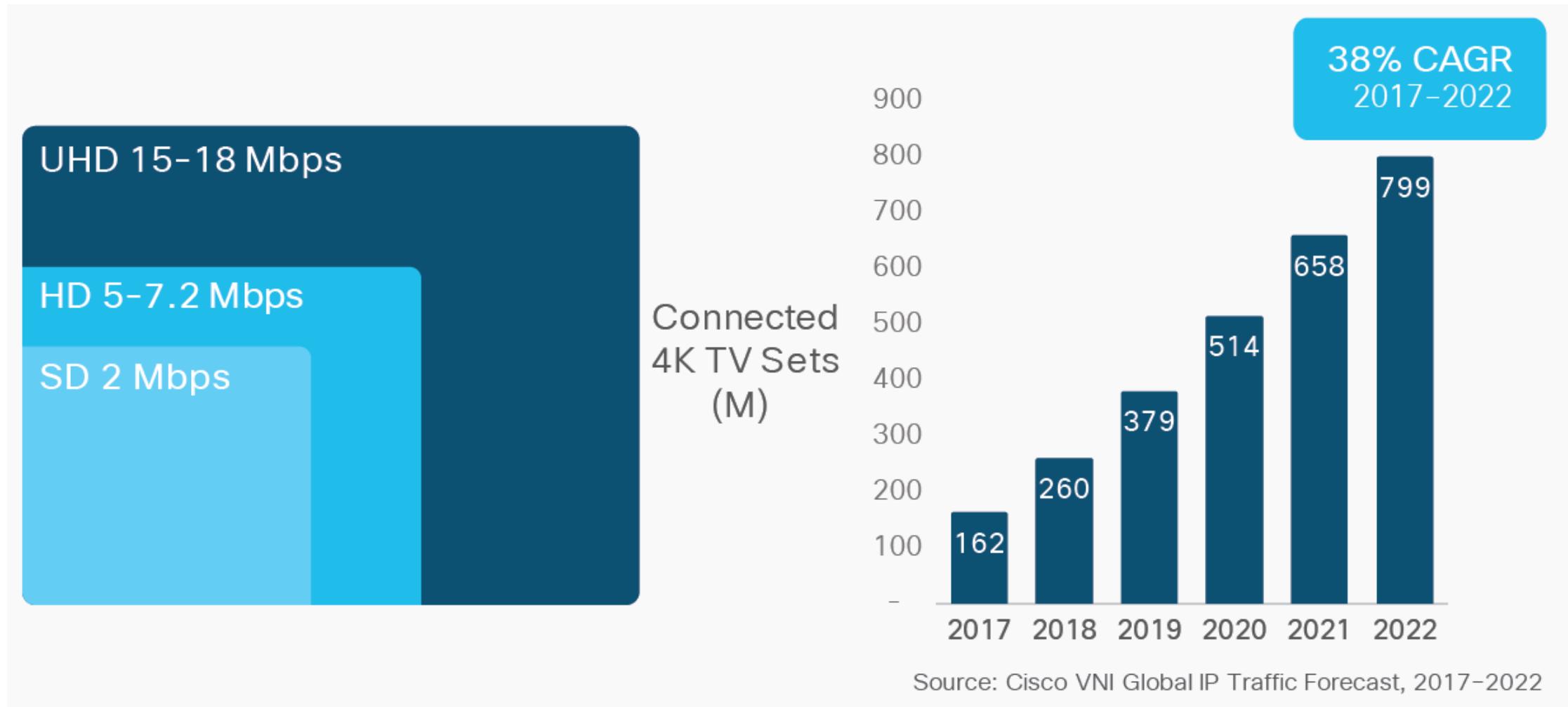
# 데이터 트래픽 증가요인 : 인터넷과 연결된 디바이스의 증가



# 데이터 트래픽 증가요인 : 디바이스별 트래픽 증가



## 데이터 트래픽 증가요인 : 동영상 화질 향상에 따른 데이터 증가



# 클라우드 컴퓨팅이 필요한 이유

# 유틸리티 (Utility) 개념

- 경제학 : 효용가치 = 재화와 용역의 사용으로부터 얻을 수 있는 주관적인 만족을 측정하는 단위
- 공리주의 철학 (Utilitarianism) : 최대다수의 최대 행복
- 도시공학 : 수도, 전기, 가스, 냉난방, 철도, 도로, 다리, 터널 등과 같은 공익 사업
- Utility software : 컴퓨터의 동작에 필수적이지는 않지만 컴퓨터를 이용하는 주 목적에 대한 부차적인 일부 특정 작업을 수행하는 소프트웨어



| 업종별 시세

업종명	전일대비	전일대비 등락현황			등락그래프				
		전체	상승	보합					
전기유틸리티	-0.66%	4	1	0	3 ■				
<hr/>									
종목명	현재가	전일비	등락률	매수호가	매도호가	거래량	거래대금	전일거래량	토론실
한전기술	19,700	▲ 250	+1.29%	19,600	19,700	119,621	2,337	118,147	<span>...</span>
한전산업	3,300	▼ 5	-0.15%	3,300	3,310	47,058	155	78,422	<span>...</span>
한전KPS	37,550	▼ 150	-0.40%	37,550	37,600	114,261	4,309	227,864	<span>...</span>
한국전력	25,350	▼ 200	-0.78%	25,350	25,400	1,346,339	34,388	1,342,841	<span>...</span>

# 인프라스트럭처 유ти리티 서비스 특징

- 가입자 누구나 사용 가능
- 필요한 때만 사용
- 품질 수준이 일정함
- 사용한 만큼 지불
- 구독 경제
- 공유 경제

Excludability

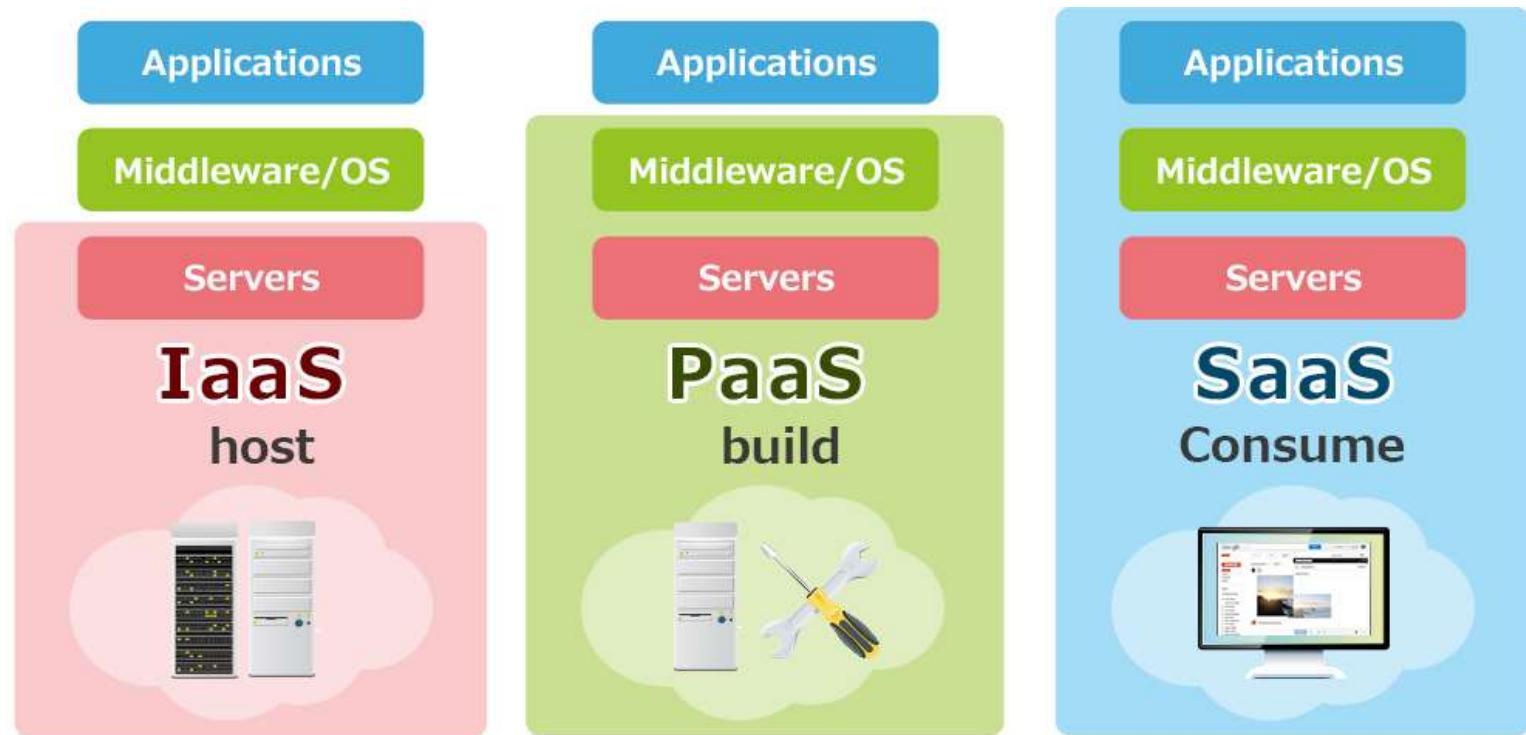
## 재화의 유형 (Types of Goods)



# 컴퓨터 서비스에서의 공유 경제

## X as a Service (XaaS)

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)
- Database as a Service (DaaS)
- Blockchain as a Service (BaaS)
- Security as a Service (SecaaS)



# 클라우드 컴퓨팅 개념의 발명

- 존 맥카티는 1950년대말~1960년대초에 클라우드 컴퓨팅의 기초 컨셉을 발명

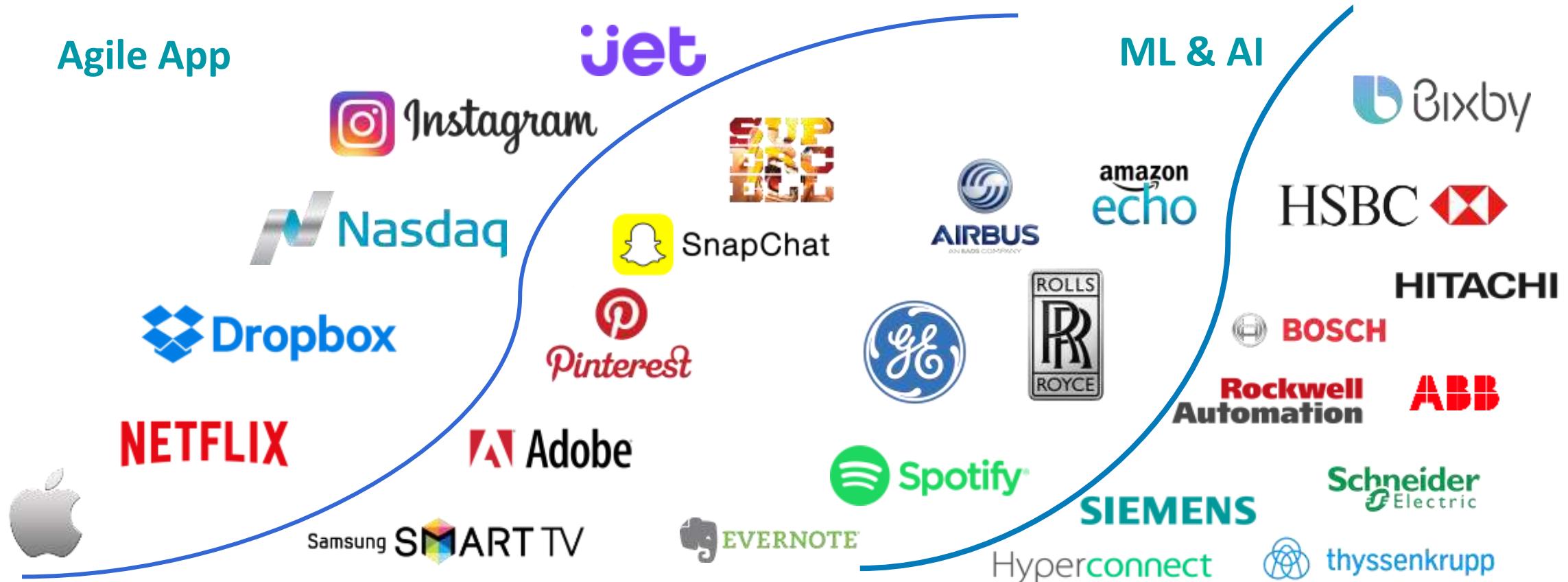
John McCarthy developed the concept of computer time-sharing in the late 1950s and early 1960s, an advance that greatly improved the efficiency of distributed computing and predicated the era of cloud computing by decades



# 클라우드 컴퓨팅으로 인한 신 경제

- Service, Not Hardware, Not Software
- No Commitment
- No Ownership
- No Capacity Provisioning
- No Intellectual Property Right
- Subscription | Stable Cash Flow
- No Protection
- No Territory Confinement
- Global Competition
- No Anti-Piracy
- Continuous Integration, Continuous Delivery
- Continuous Sales

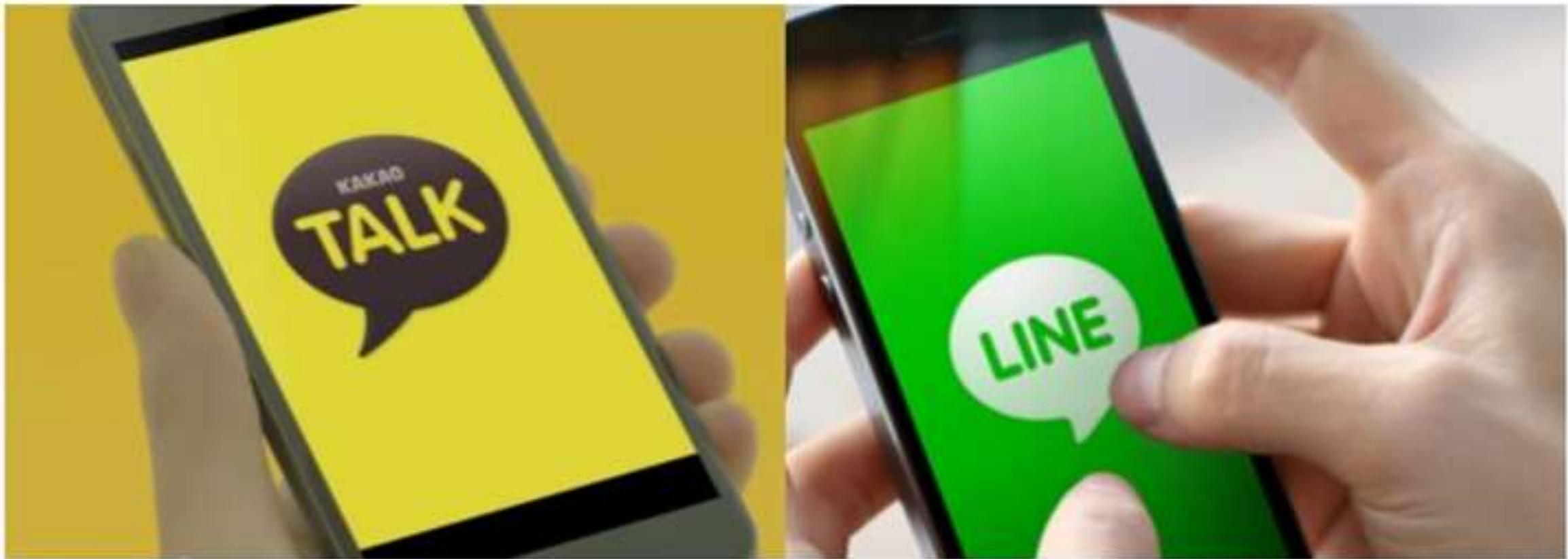
# 클라우드 전환의 물결



# **클라우드 컴퓨팅 수요 요인**

## **(Cloud Demand Drivers)**

## 클라우드 수요요인 – Mobile First (1)



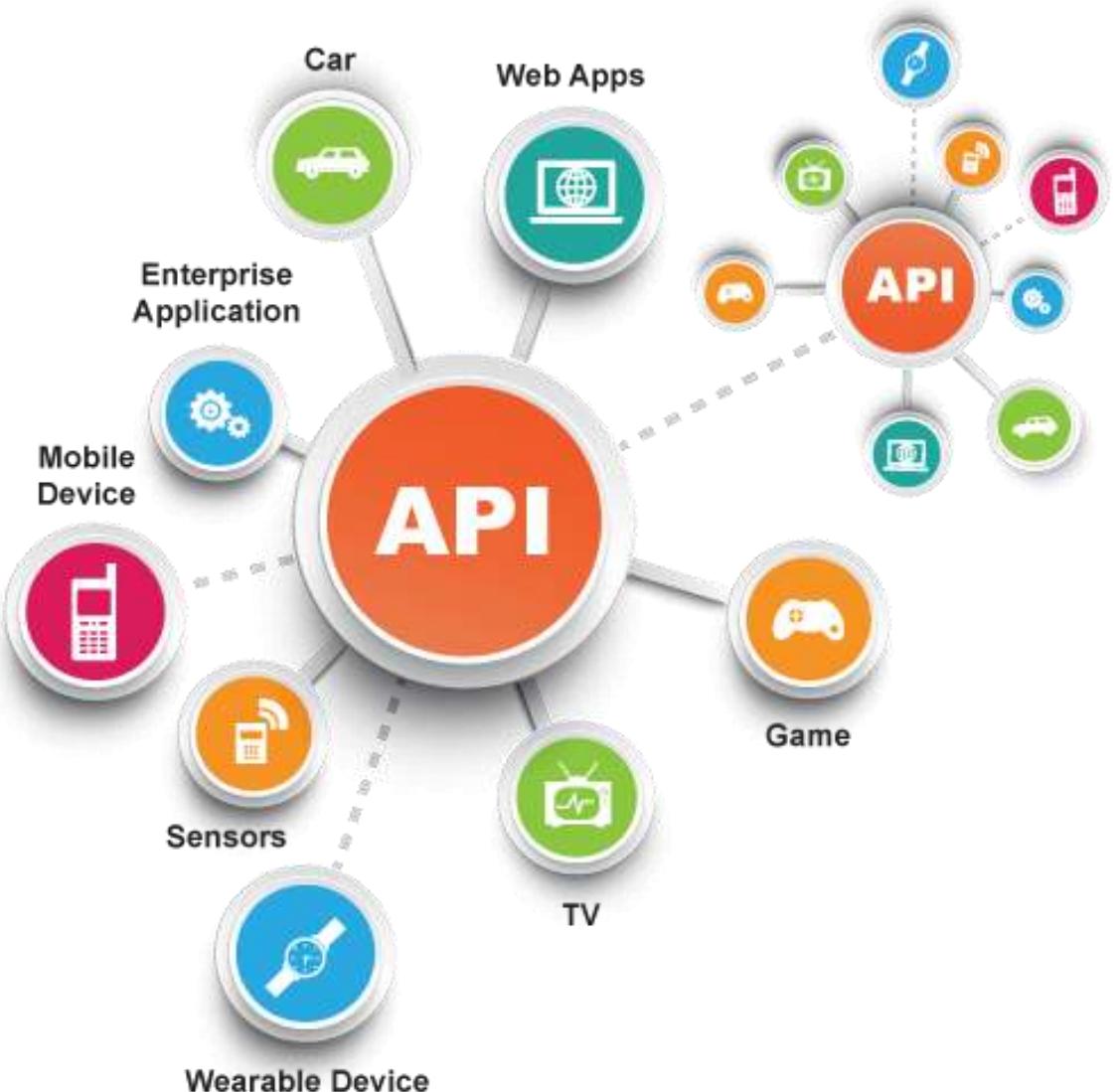
# 클라우드 수요요인 – Mobile First (2)



# 클라우드 수요요인 – Big Data

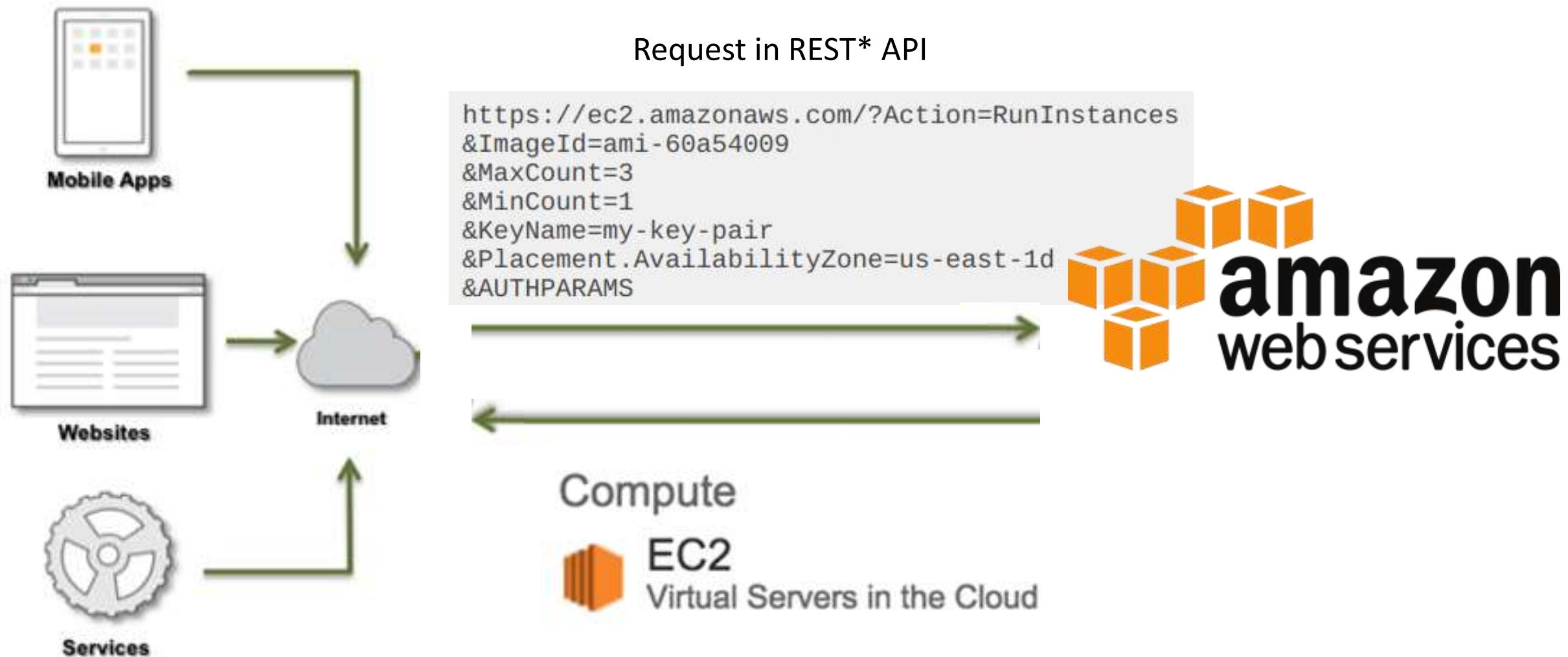


# 클라우드 수요요인 – API Service의 발달 (1)



# 클라우드 수요요인 – API Service의 발달 (2)

## IAAS API CALL/REQUEST RETURNS INFRASTRUCTURE

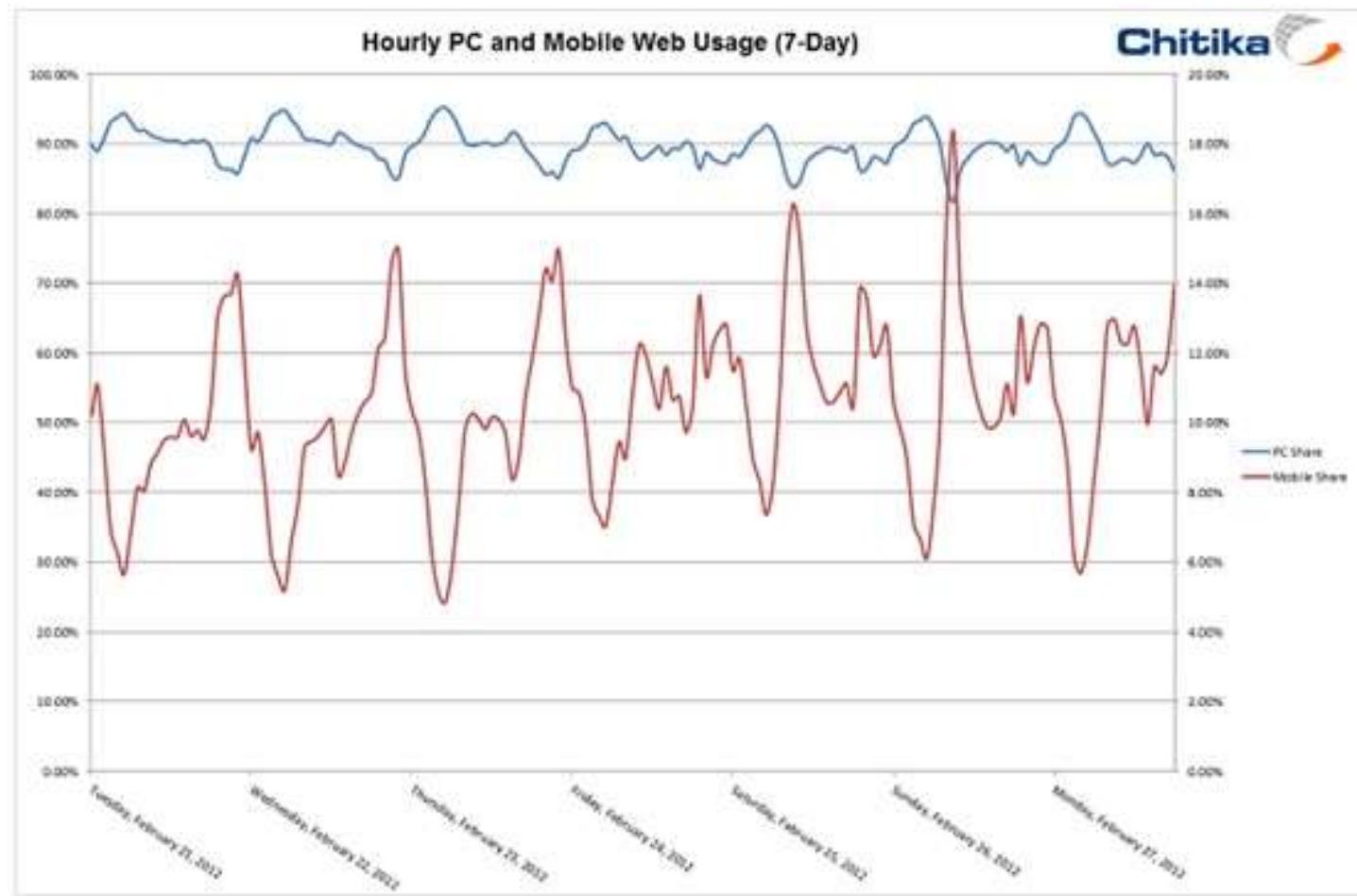


# 클라우드 수요요인 – IoT / M2M 통신 증가

- IOT IS A BI-DIRECTIONAL DATA SERVICE IN THE CLOUD



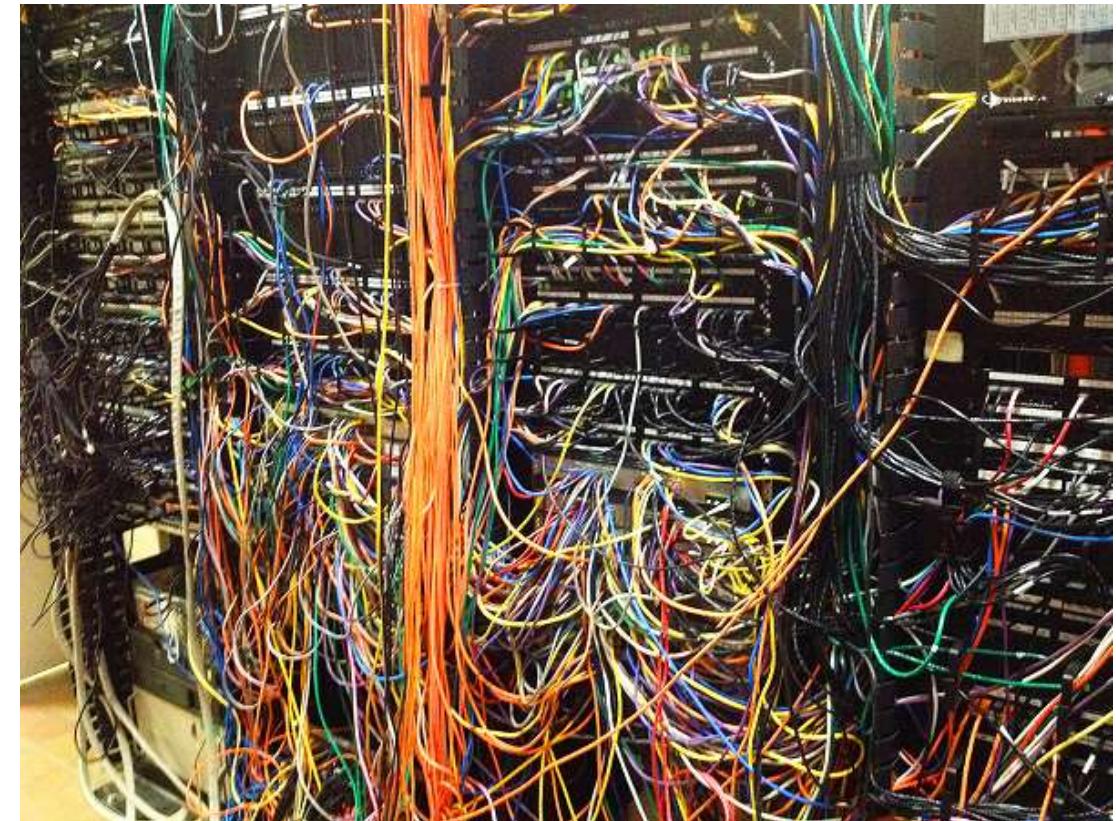
# 클라우드 수요요인 – Traffic 예측의 불확실성



# 클라우드 수요요인 – Data Center Capacity 관리의 어려움



vs.



Google Data Center, [https://services.google.com/fh/files/misc/google\\_2019-environmental-report.pdf](https://services.google.com/fh/files/misc/google_2019-environmental-report.pdf)

Extreme Tech, <https://www.extremetech.com/extreme/187954-brace-for-the-bgpocalypse-big-disruptions-loom-as-internet-overgrowth-continues>

# 클라우드 수요요인 – 인공지능 (1)

## Google DeepMind's AlphaGo

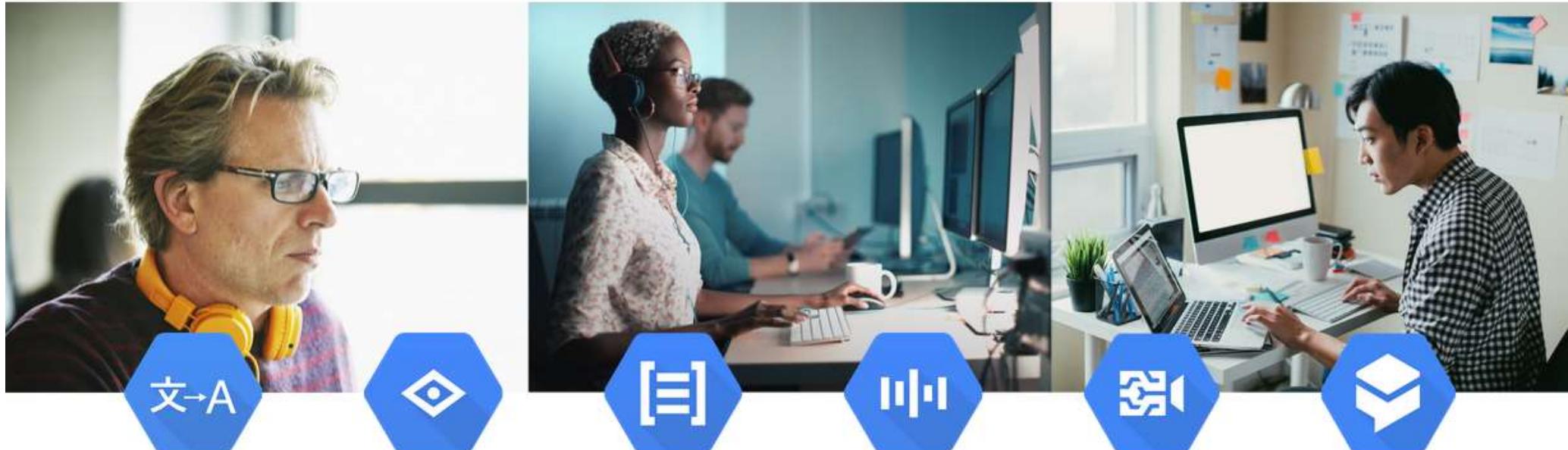


Google DeepMind



# 클라우드 수요요인 – 인공지능 (2)

- 클라우드에서 기본으로 제공하는 기계학습 엔진



# **클라우드 컴퓨팅과 디지털 변혁**

## **(Digital Transformation)**

# 디지털 변혁을 위한 기업들의 고민



## The CEO Interview

“Industrial companies are in the information business whether they want to be or not.”

—Jeff Immelt





“Machine learning. This is the next transformation... the programming paradigm is changing. Instead of programming a computer, you teach a computer to learn something and it does what you want.”



Eric Schmidt, Google Chairman of the Board



“Machine learning and artificial intelligence (AI) as core component to the future of digital transformation in businesses”

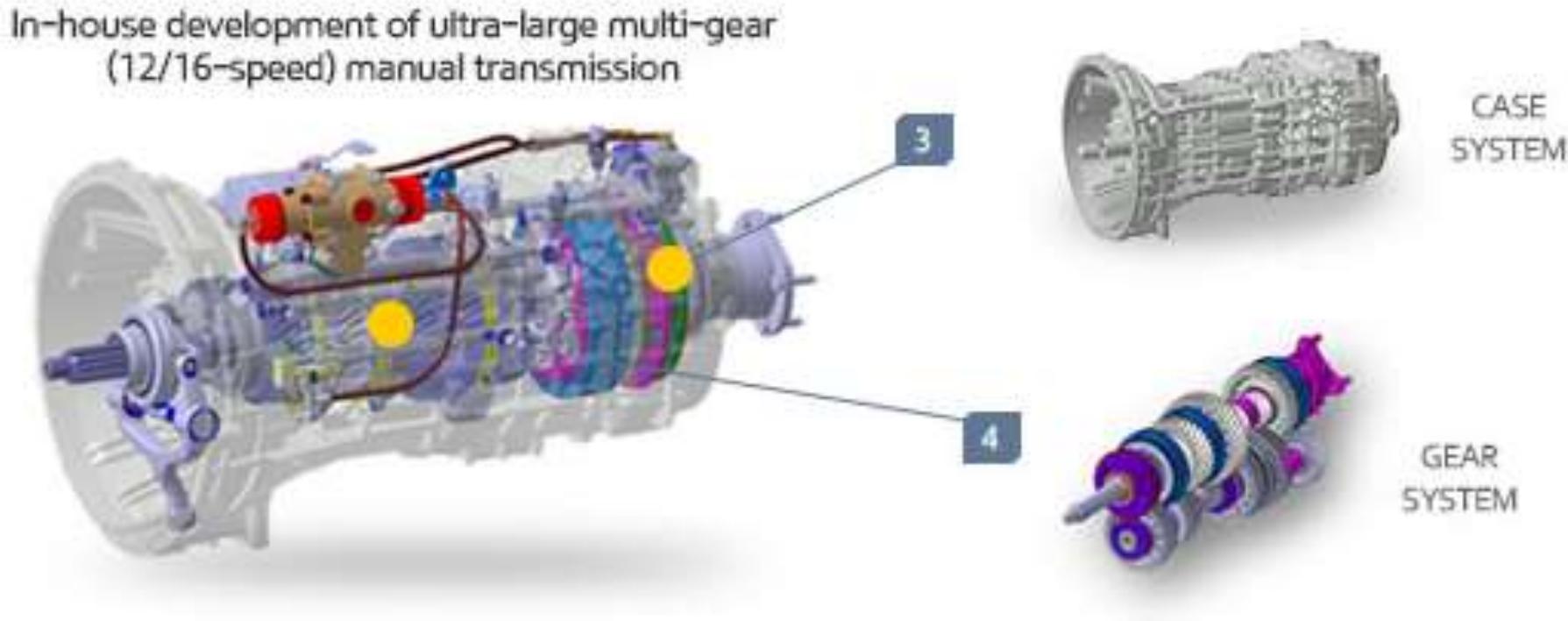
“Every business is going to have a bot interface”



Satya Nadella, Microsoft CEO

# 클라우드 컴퓨팅을 이용한 R&D – 시뮬레이션 / 설계

Due to the flexible scalability of the cloud computing, computation intensive R&D time can easily be shortened.



# 클라우드 컴퓨팅을 이용한 R&D – 화성 탐사

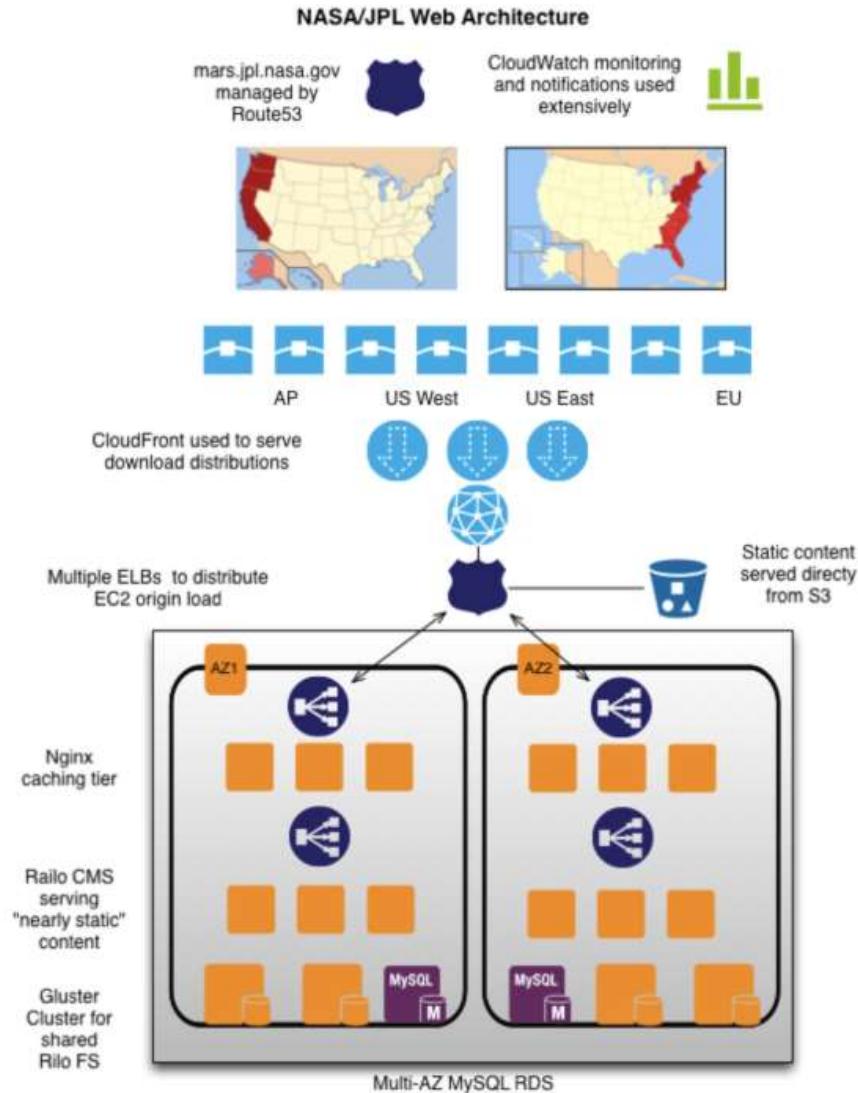
## Curiosity - The Next Mars Rover



[https://www.nasa.gov/mission\\_pages/msl/multimedia/gallery/pia14156.html](https://www.nasa.gov/mission_pages/msl/multimedia/gallery/pia14156.html)

<https://aws.amazon.com/ko/solutions/case-studies/nasa-jpl-curiosity/>

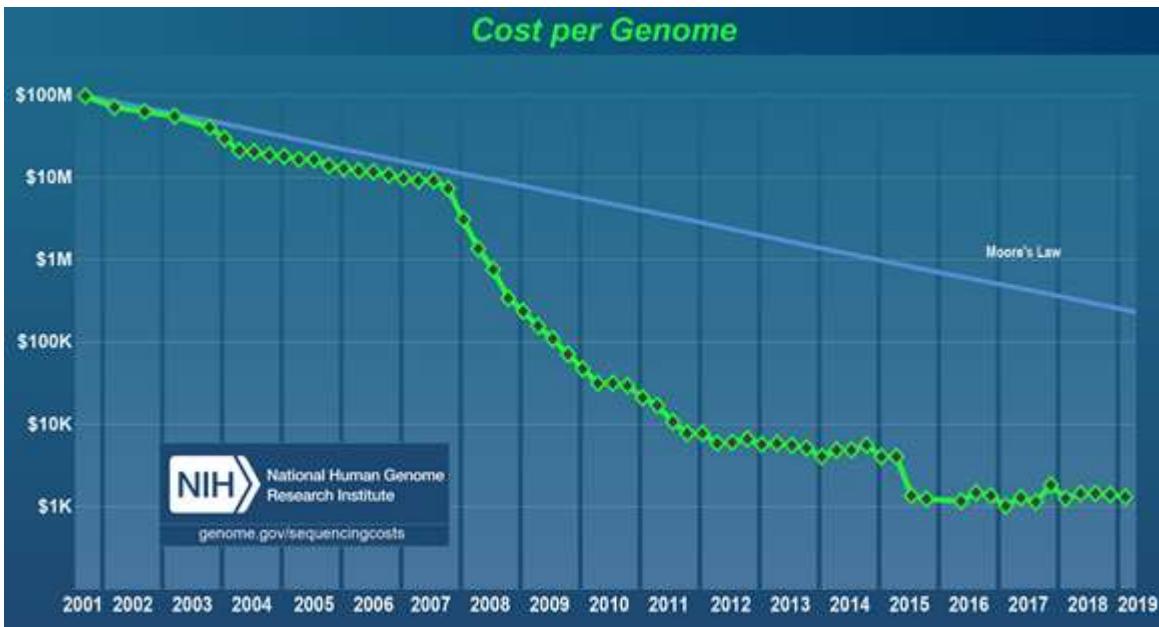
## NASA/JPL Web Architecture



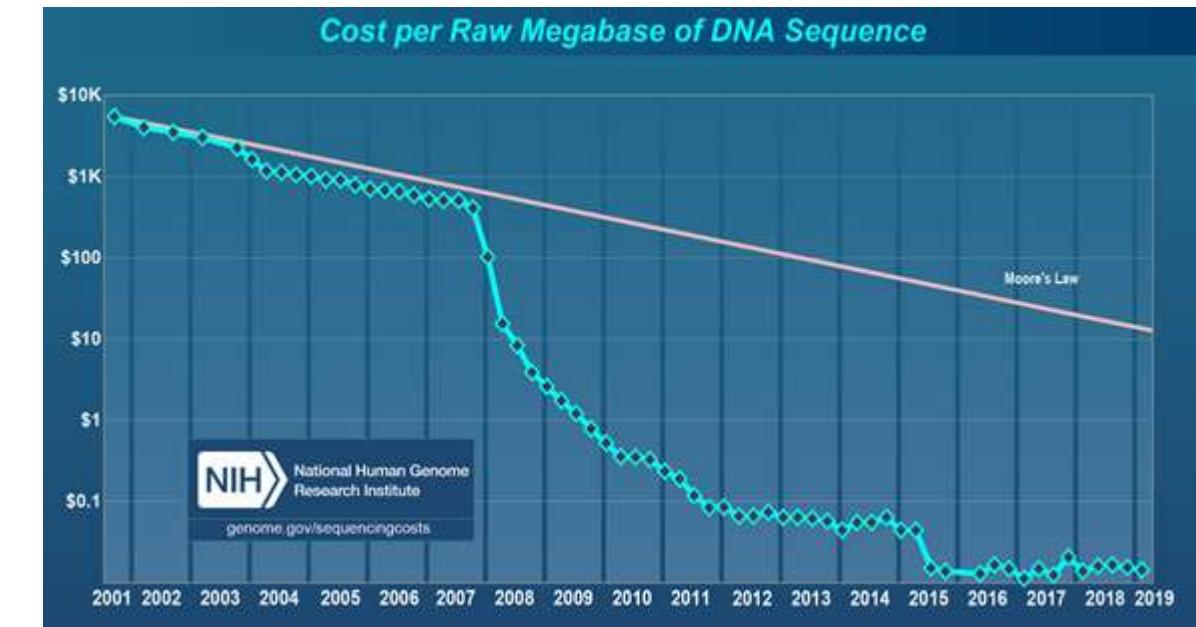
# 클라우드 컴퓨팅을 이용한 R&D

## 클라우드 데이터 분석의 혁신성 : 인간 유전자 분석의 사례

Sequencing cost per genome - 2019



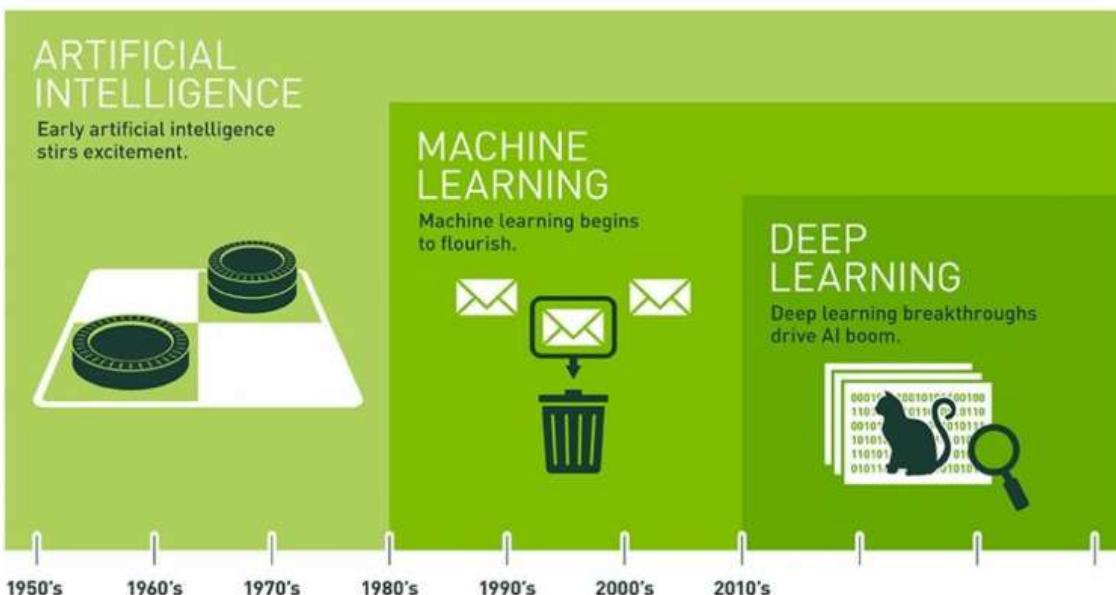
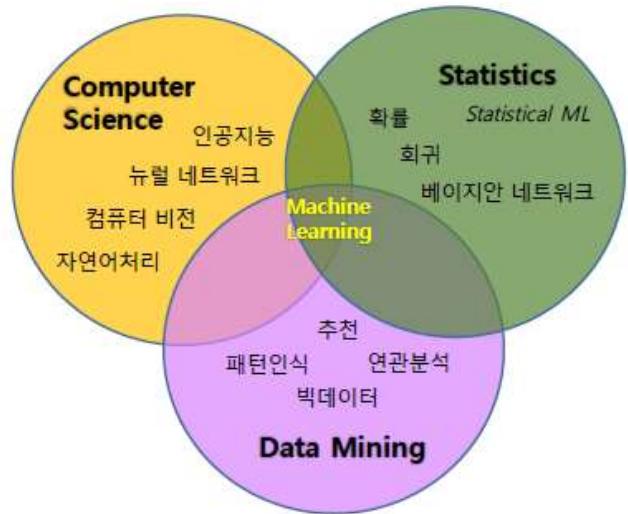
Sequencing cost per Mb - 2019



Source: NIH

<https://www.genome.gov/about-genomics/fact-sheets/DNA-Sequencing-Costs-Data>

# 기계학습 (Machine Learning) vs. 인공지능



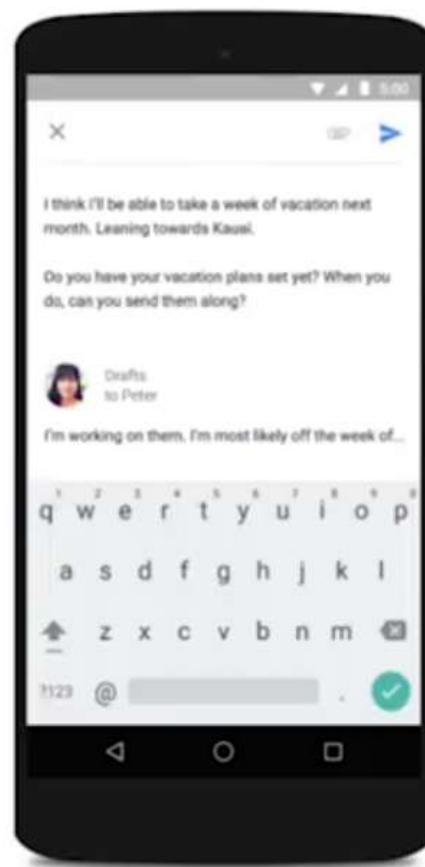
## ■ Machine Learning Application

- 많은 수동 조정과 규칙이 필요한 문제
- 해결하기 너무 어렵거나 알려진 해가 없는 문제
  - 음성인식('one' vs 'two')
  - 얼굴인식(눈, 코, 입의 위치)
- 변화하는 환경에 적응해야 하는 문제
- 복잡한 문제와 대량의 데이터에서 통찰 얻기(데이터 마이닝)

nVidia.

Aurélien Géron (2017), 박해선 역, "Hands-On Machine Learning with Scikit-Learn and TensorFlow"

Machine Learning is not new, but it is now mainstream



Search

People who bought ...

Spam filtering

Suggest next video

Route planning

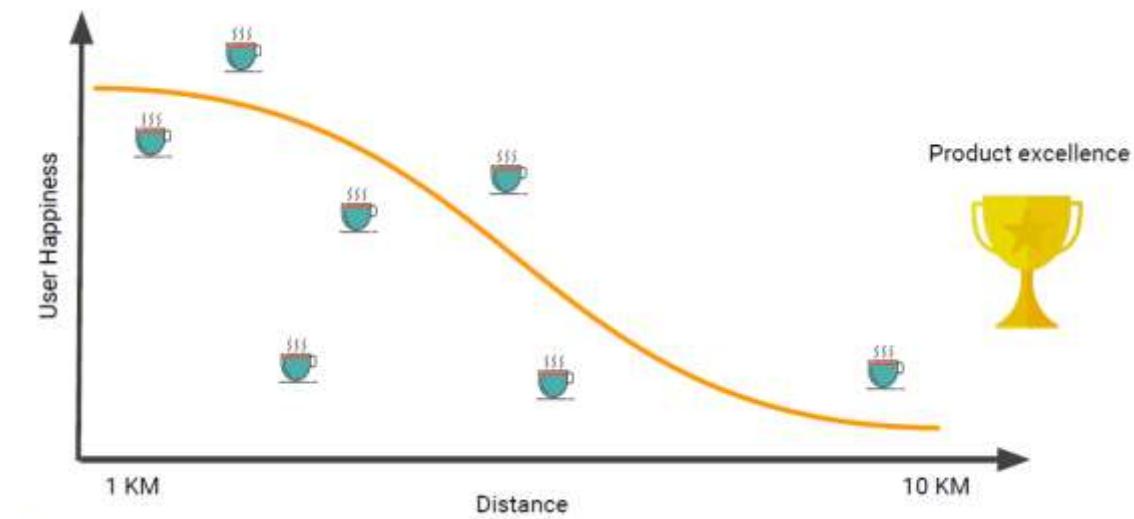
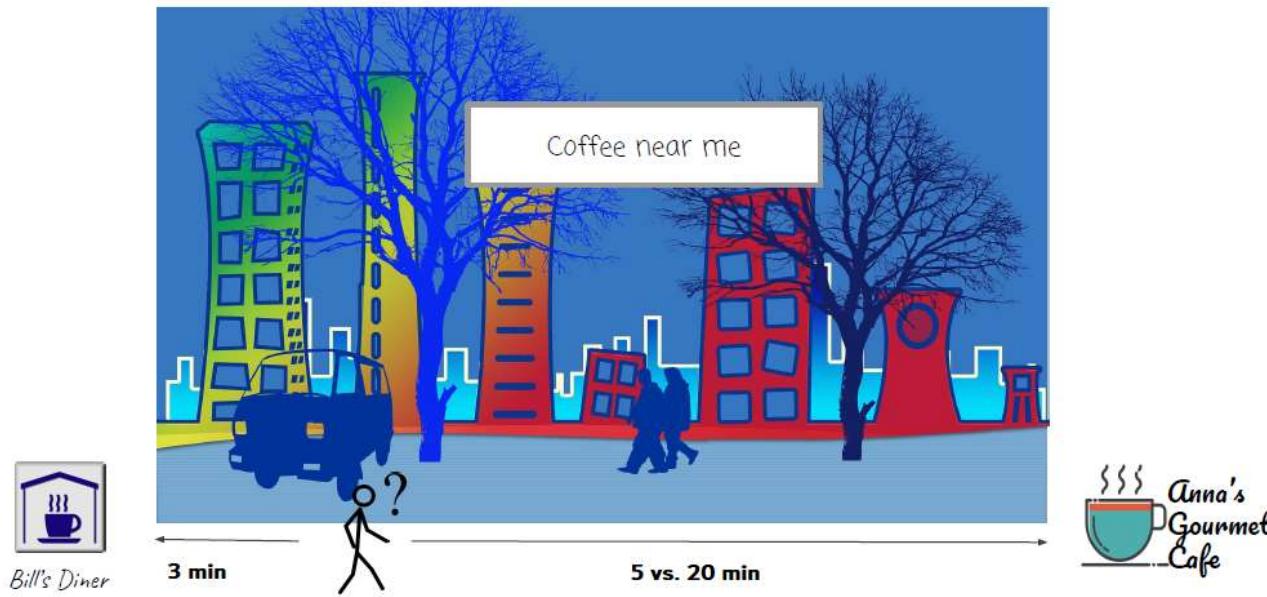
Smart Reply



What's common to all of these use cases of Machine Learning?

# 기계학습 활용의 예

ML converts examples into knowledge

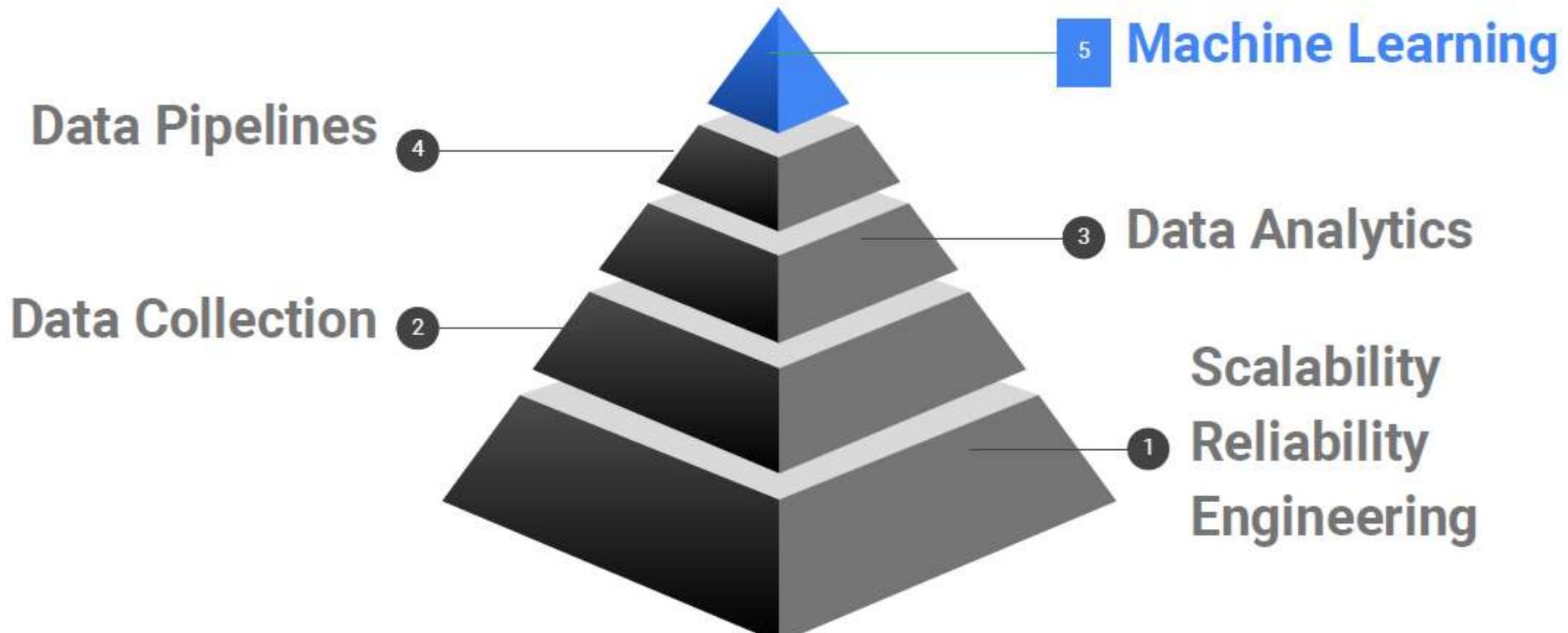


# 데이터 수집과 기계학습

## ML Effort Allocation

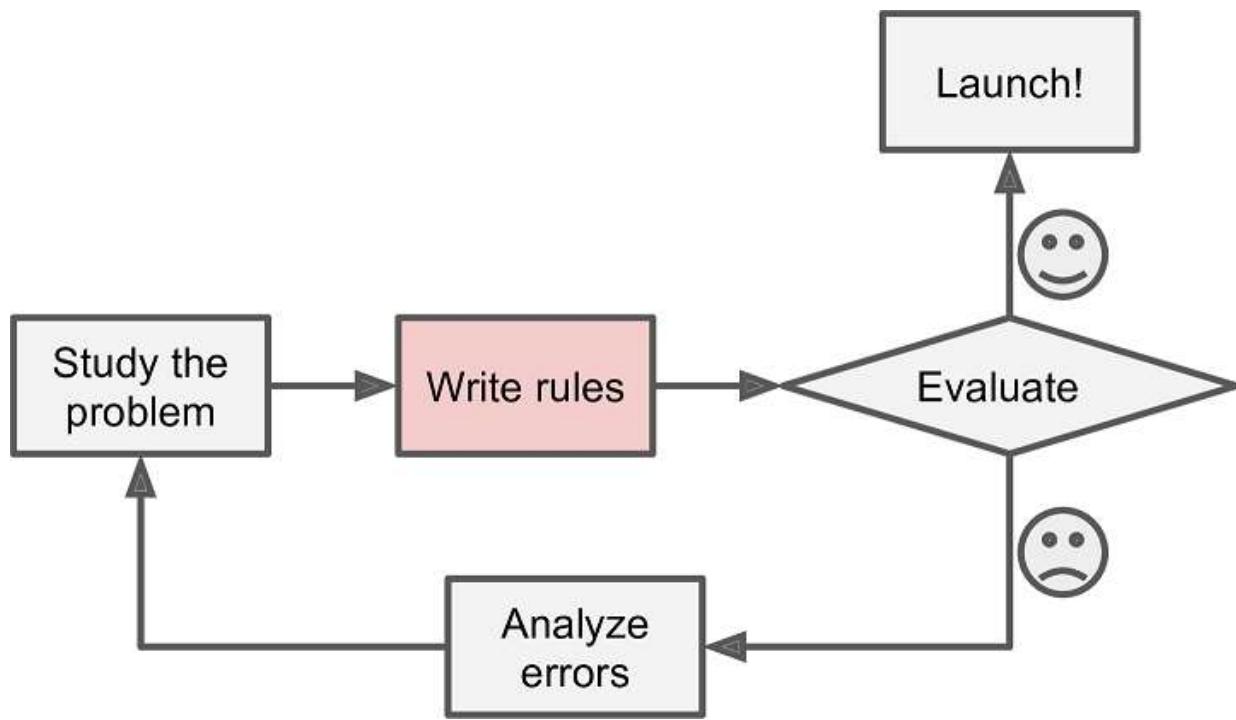


# 데이터 엔지니어 vs. 데이터 과학자

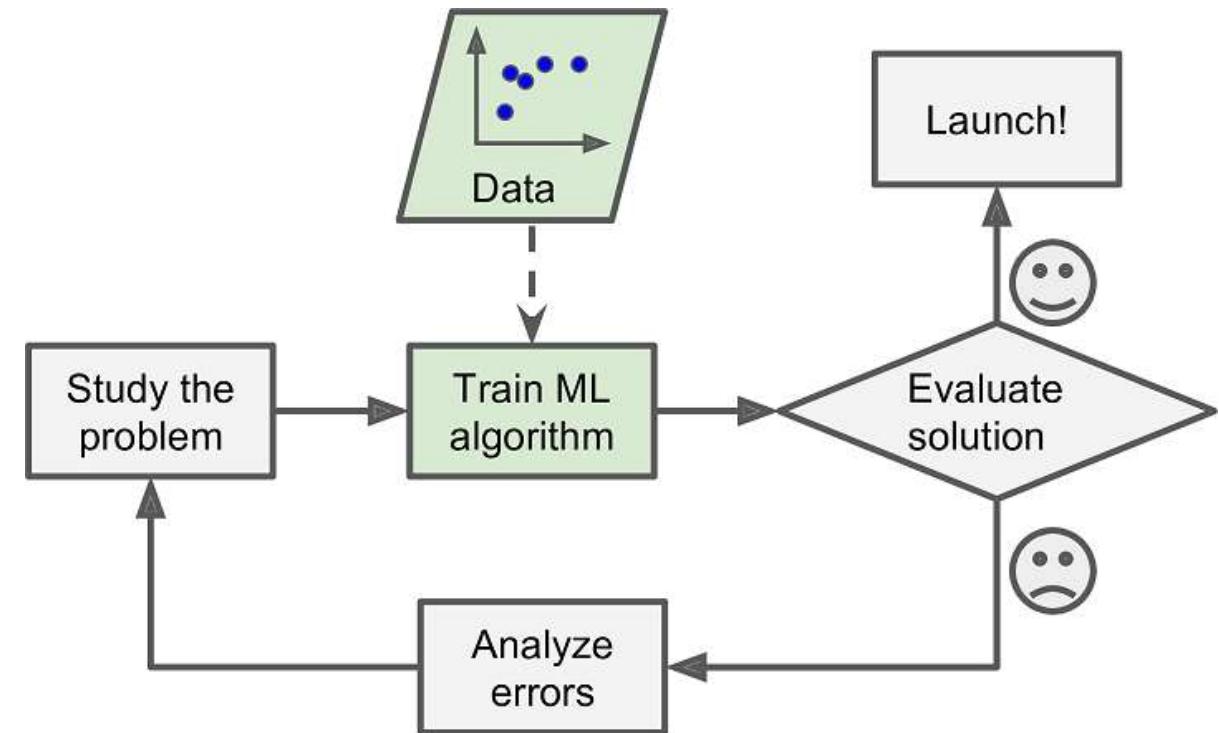


# 전통적 방법 vs 기계학습 방법

- Traditional Approach



- Machine Learning Approach



# 시사점

# THE FUTURE OF EMPLOYMENT: HOW SUSCEPTIBLE ARE JOBS TO COMPUTERISATION?\*

Carl Benedikt Frey<sup>†</sup> and Michael A. Osborne<sup>‡</sup>

September 17, 2013

**DR CARL BENEDIKT FREY**

Oxford Martin Citi Fellow



**MICHAEL A  
OSBORNE**



Dyson Associate Professor in Machine Learning at the University of Oxford

<https://www.oxfordmartin.ox.ac.uk/publications/the-future-of-employment/>  
[https://www.oxfordmartin.ox.ac.uk/downloads/academic/The\\_Future\\_of\\_Employment.pdf](https://www.oxfordmartin.ox.ac.uk/downloads/academic/The_Future_of_Employment.pdf)  
<https://www.robots.ox.ac.uk/~mosb/>

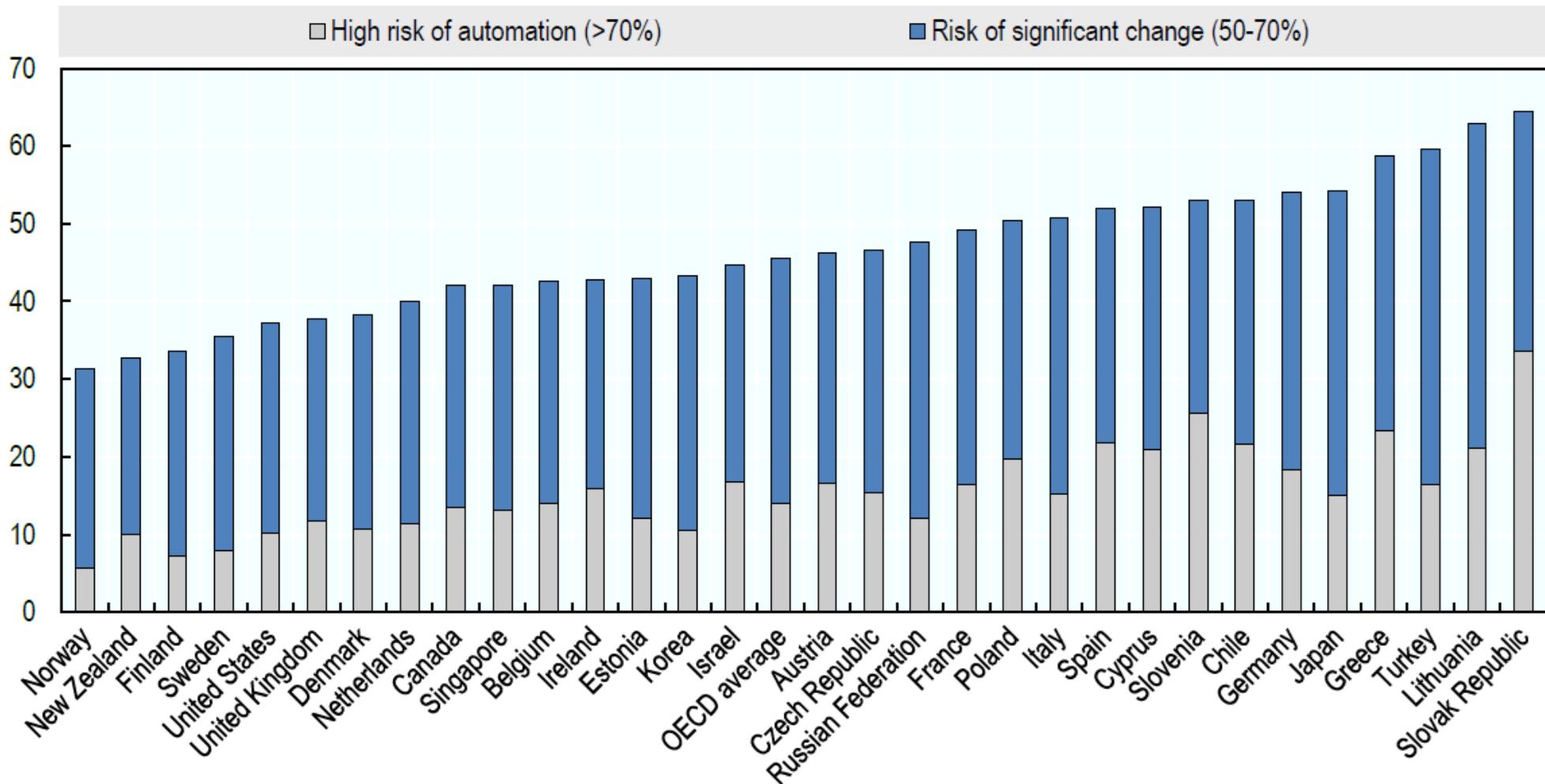
Rank	Probability	Occupation
1	0.0028	Recreational Therapists
14	0.0041	Sales Eng
32	0.0065	Com Sys Analysts
56	0.012	Microbiologist
77	0.017	Chem Eng
87~88	0.021	Mater Sci/Eng
99	0.027	Biochem, Biophys
619	0.95	Animal Breeders
629	0.96	Office Clerks, General
641	0.96	Cooks, Restaurant
674	0.98	Driver/Sales Workers
676	0.98	Parts Salespersons
698	0.99	Insurance Underwriters
702	0.99	Telemarketers

# AI and robots will destroy fewer jobs than previously feared, says new OECD report

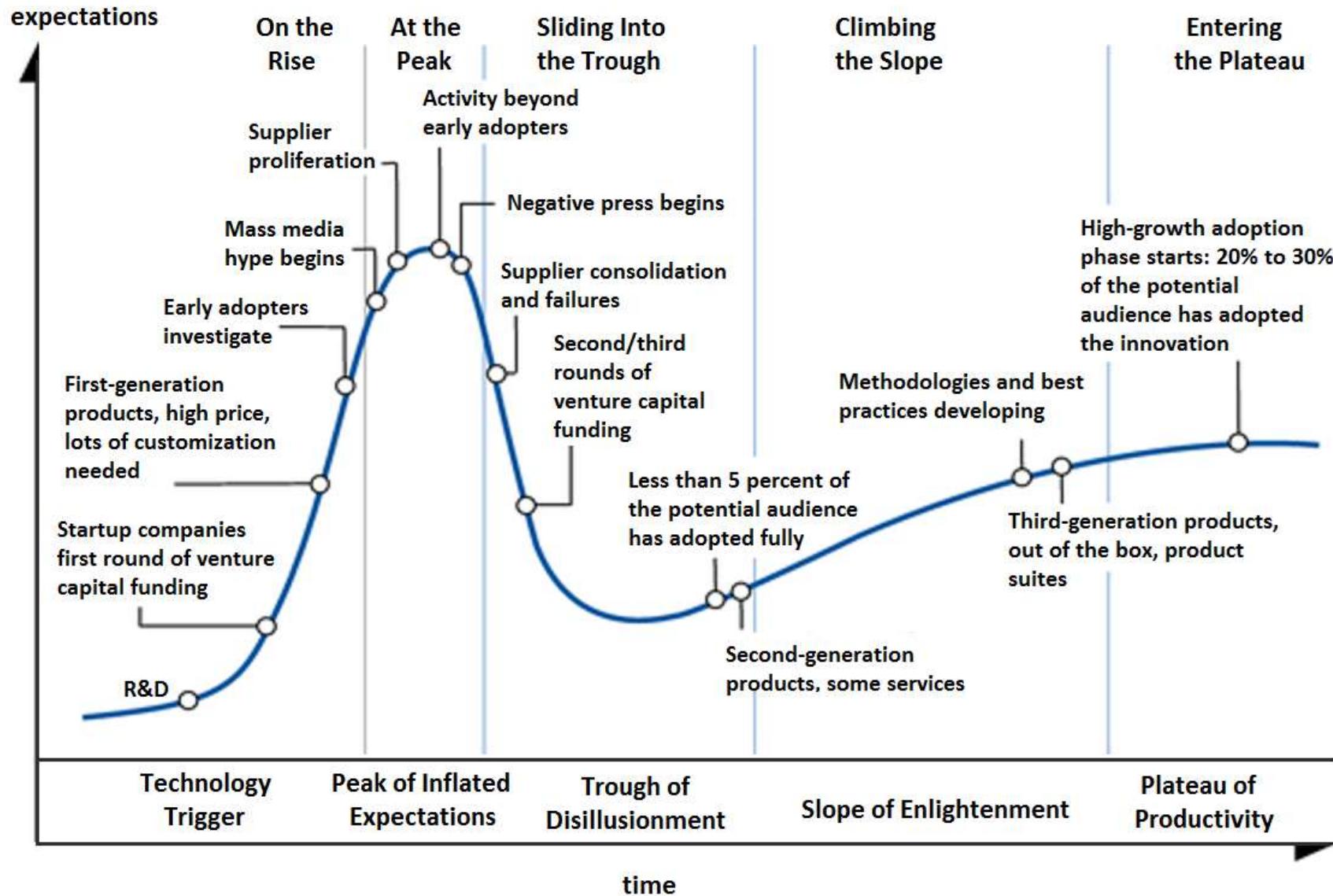
THE VERGE

Apr 3, 2018

*But the impact will still be significant, increasing societal division between the rich and the poor*



# Gartner Hype Cycle Curve

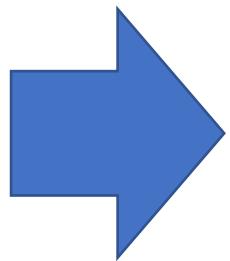


**"It is not the strongest of the species that survives, not the most intelligent that survives. It is the one that is the most adaptable to change."**

**Charles Darwin**

# Current Old Educational System Can NOT Catch the Fast Development of Technology

Medieval university



Renaissance Tutoring

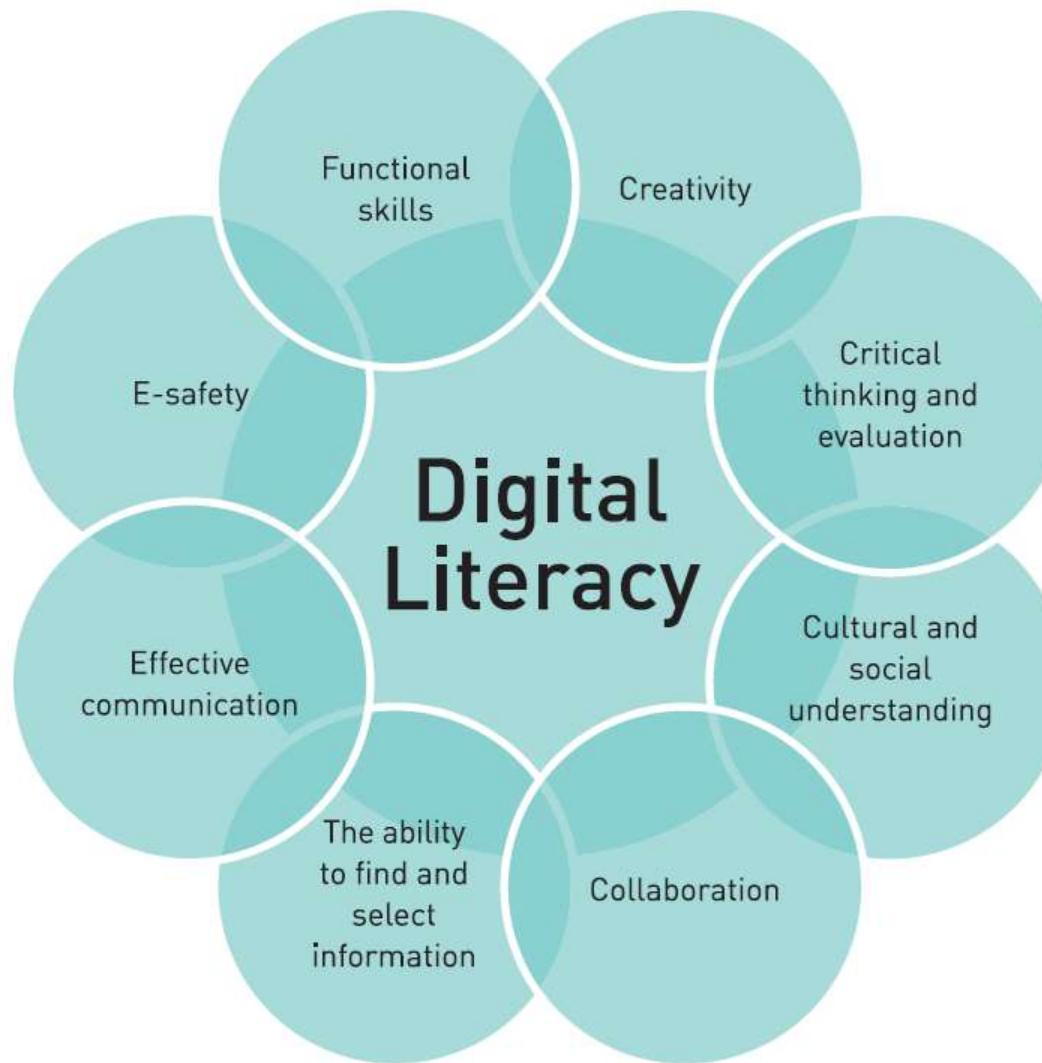


[https://en.wikipedia.org/wiki/Medieval\\_university](https://en.wikipedia.org/wiki/Medieval_university)

<https://www.thegreatcoursesdaily.com/education-in-the-renaissance/>

# Digital Literacy

문맹 → 컴맹 → 넷맹 → 폰맹 → 클맹 ?, 인맹 ?



Hague & Payton, *Digital literacy across the curriculum*, FutureLab (2010)

# The top 5 hard skills companies need most in 2019

Based on research from LinkedIn Learning

1. Cloud Computing
2. Artificial Intelligence
3. Analytical Reasoning
4. People Management
5. UX Design

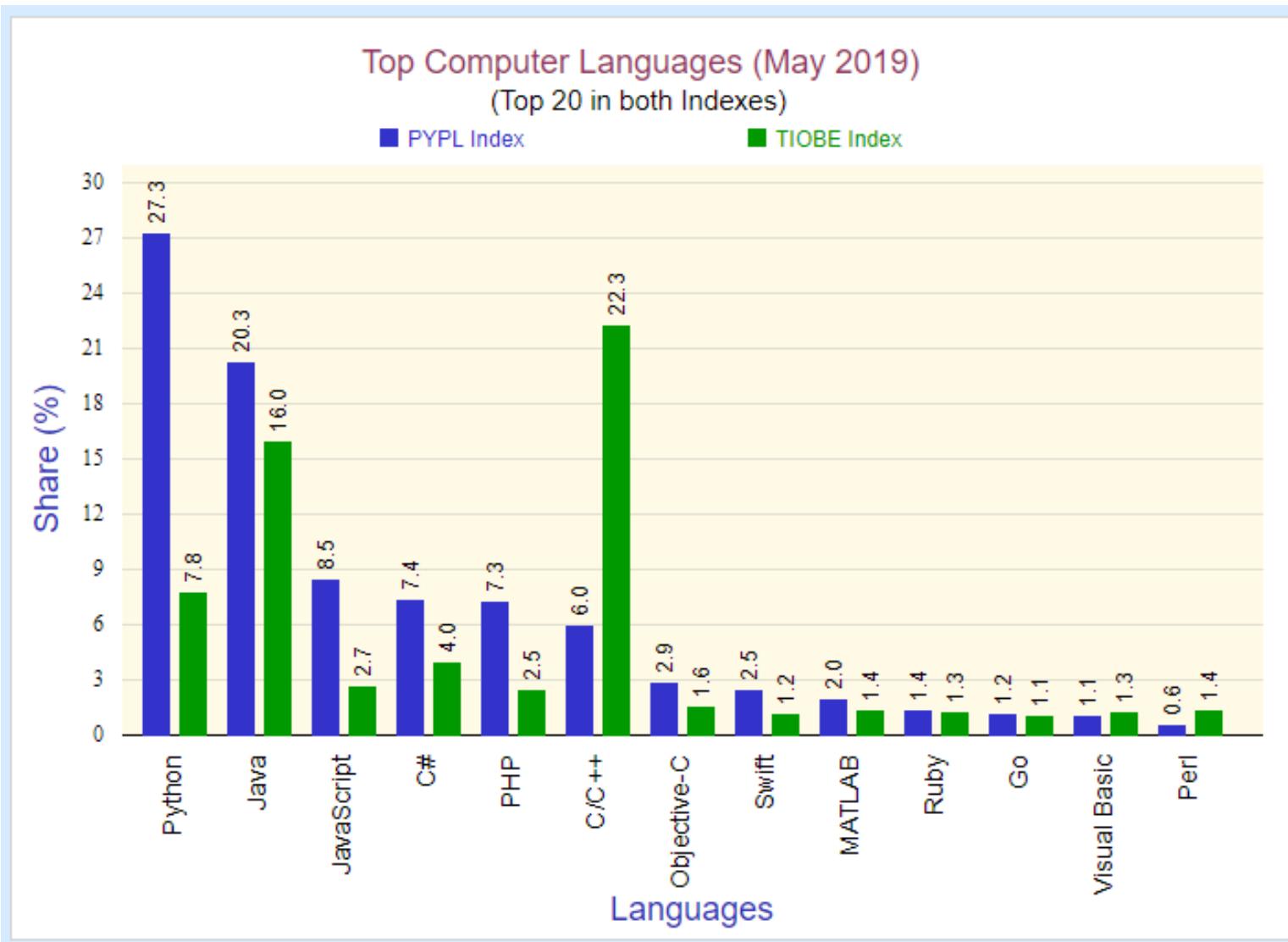
Source: LinkedIn

# 15 Top Paying IT Certifications In 2019

Most Valuable IT Certifications, 2019 (Source: Global Knowledge Study, 15 Top-Paying Certifications for 2019)	
Certification	Annual Salary
1. Google Cloud Certified Professional Cloud Architect	\$ 139,529
2. PMP® - Project Management Professional	\$ 135,798
3. Certified ScrumMaster	\$ 135,441
4. AWS Certified Solutions Architect - Associates	\$ 132,840
5. AWS Certified Developer – Associate	\$ 130,369
6. MCSE: Server Infrastructure	\$ 121,288
7. ITIL® Foundation	\$ 120,566
8. CISM - Certified Information Security Manager	\$ 118,412
9. CRISC - Certified in Risk and Information Systems Control	\$ 117,395
10. CISSP - Certified Information Systems Security Professional	\$ 116,900
11. CEH - Certified Ethical Hacker	\$ 116,306
12. Citrix Certified Associate - Virtualization (CCA-V)	\$ 113,442
13. Security+	\$ 110,321
14. Network+	\$ 107,143
15. CCNP Routing and Switching	\$ 106,957

Forbes

# Top Computer Languages



Statistics Times

# Start Right Now, It's Free !



[www.kocw.net](http://www.kocw.net)



<https://www.coursera.org/>



[www.kmooc.kr](http://www.kmooc.kr)

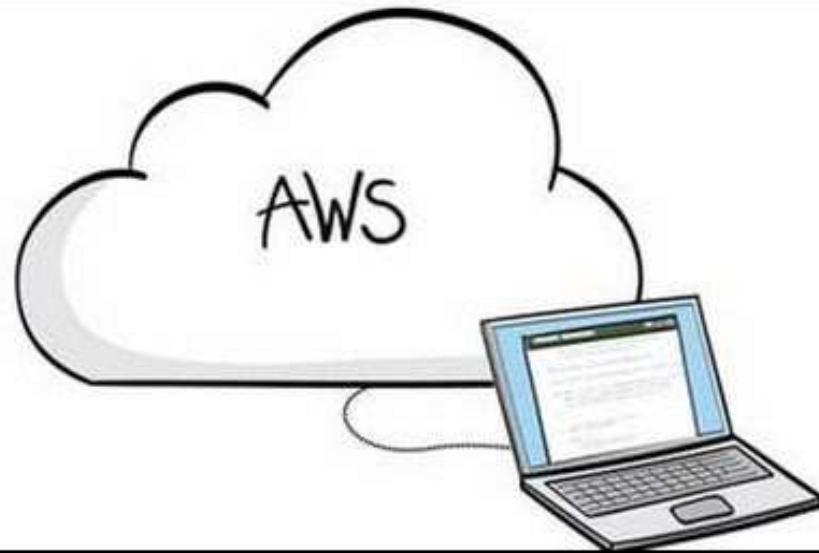


<https://www.youtube.com>

## **Part II**

# **클라우드 컴퓨팅 핵심 개념**

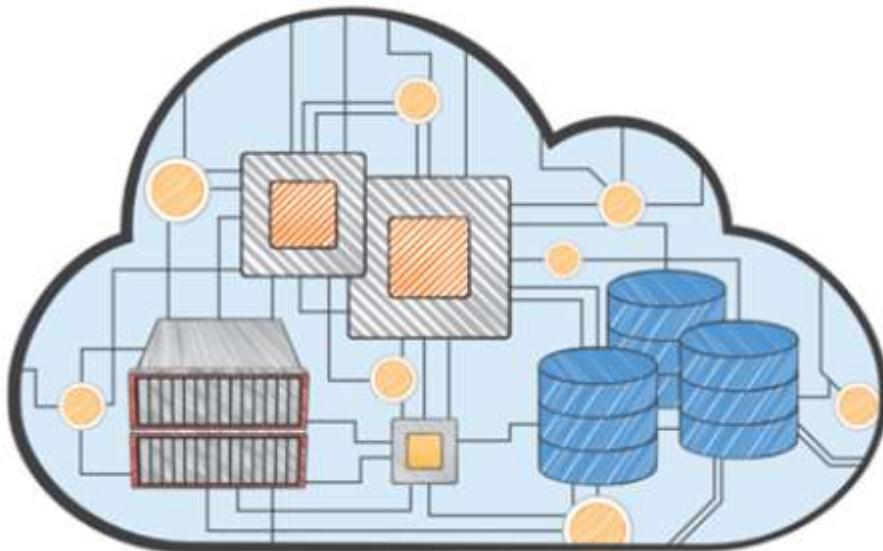
# 클라우드 컴퓨팅 : What & Why ?



Why are customers adopting  
cloud computing?



# 퍼블릭 클라우드 컴퓨팅 특징



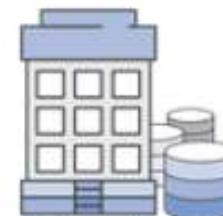
- 온디맨드
- IT 리소스
- 온라인에서 접속 가능
- 종량 과금제

# 퍼블릭 클라우드 장점 : 아마존의 설명



Amazon Web Services

- 초기 투자금 없음
- 낮은 유지 비용
- 혁신에 집중
- 유연한 혁신
- 속도 및 민첩성
- 온디맨드 글로벌 접근성



온프레미스 데이터 센터

- 대규모 실행 자본 지출
- 인력, 패치 및 업그레이드 주기
- 시스템 관리
- 고정 용량
- 조달 및 설치
- 한정된 지리적 리전

## 퍼블릭 클라우드 장점 : 구글의 설명



### On-demand self-service

No human intervention needed to get resources



### Broad network access

Access from anywhere



### Resource pooling

Provider shares resources to customers



### Rapid elasticity

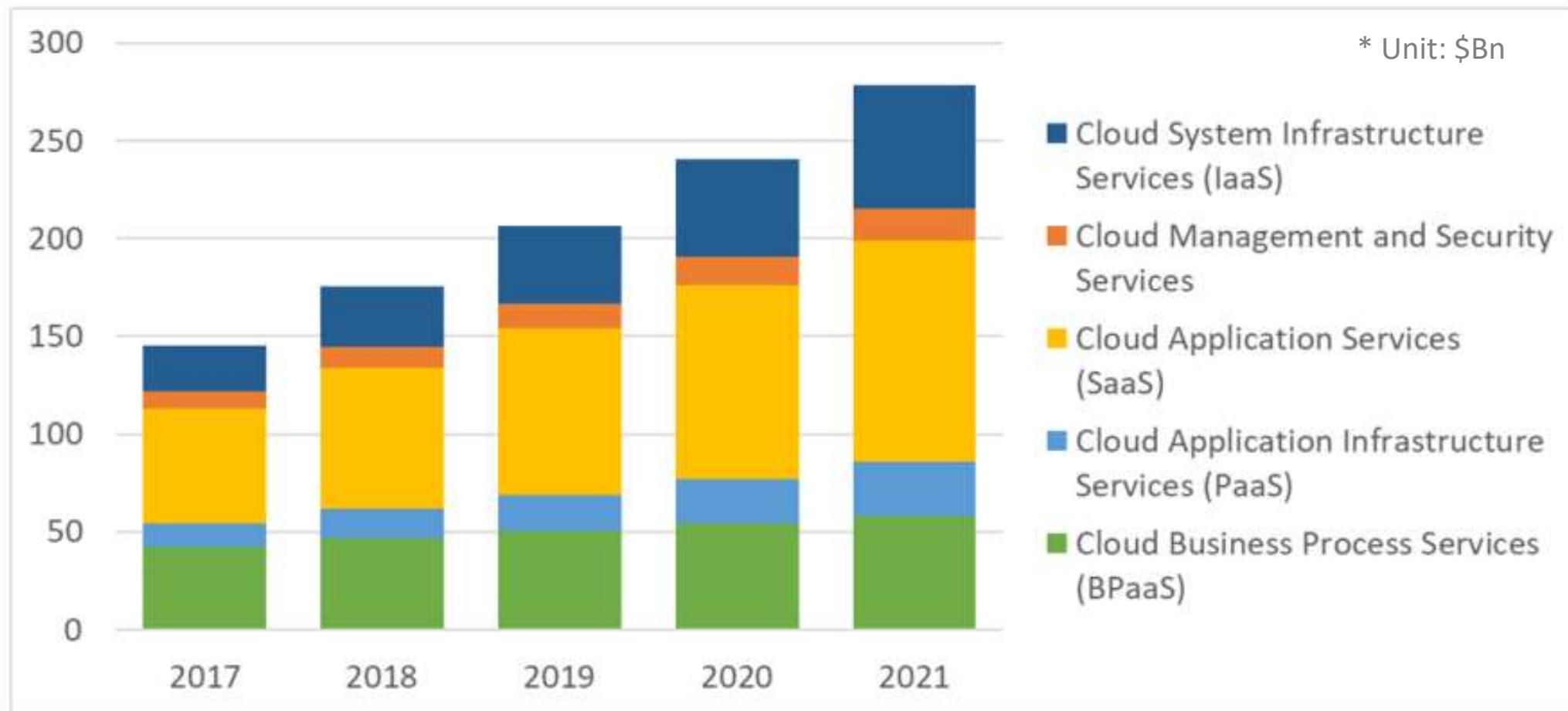
Get more resources quickly as needed



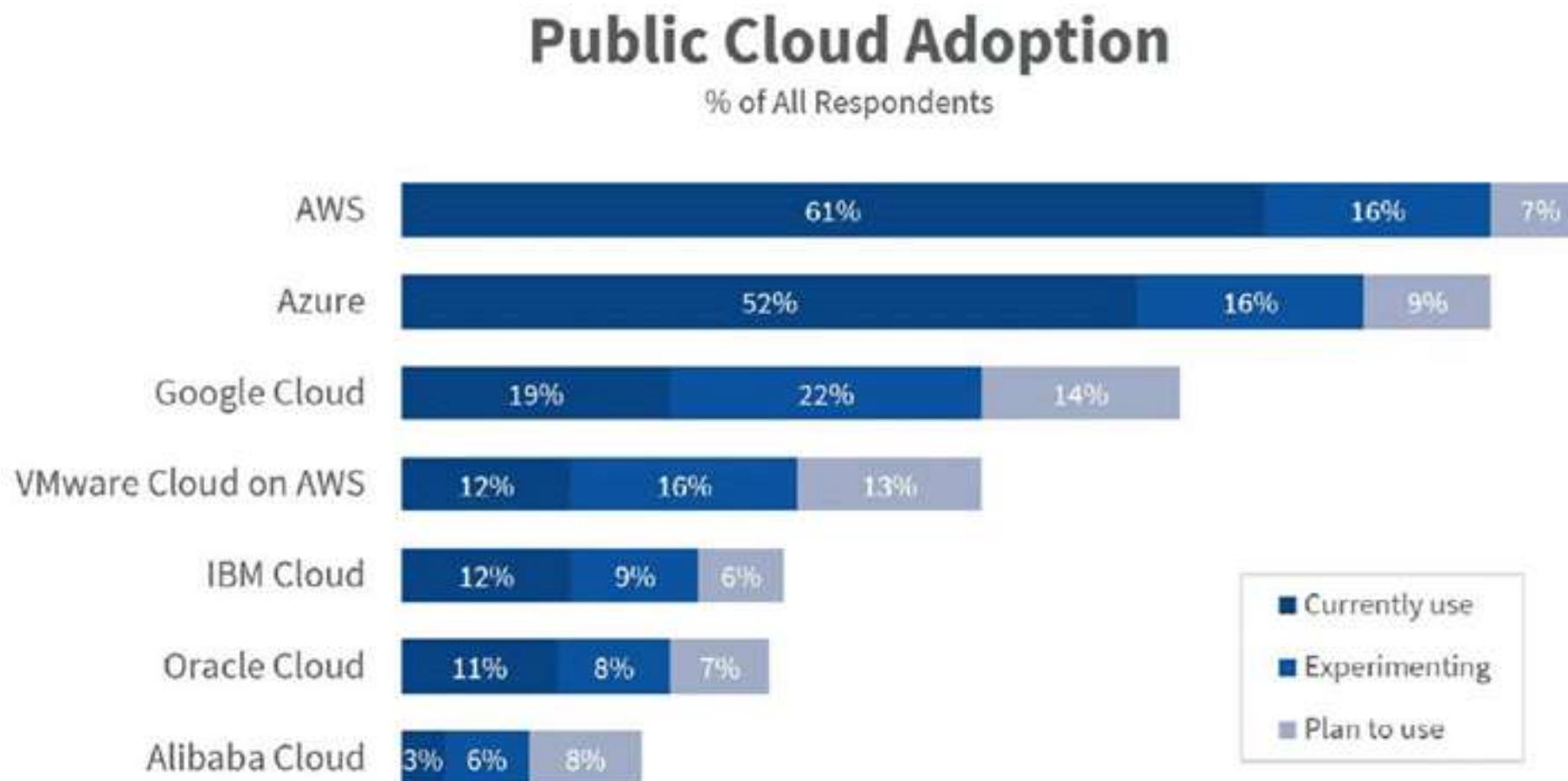
### Measured service

Pay only for what you consume

# 퍼블릭 클라우드 시장 전망



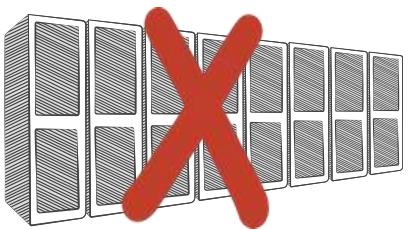
## 주요 퍼블릭 클라우드 업체



# 왜 Cloud Computing 인가? – 6가지 이점

## 초기 선투자 비용 없음

고정비용을 가변비용으로 대체  
미리 서버를 구매할 필요 없음



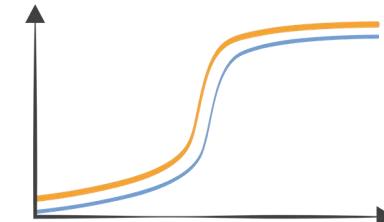
## 운영 비용 절감

사용한 만큼만 지불하며 규모의 경제로 인한  
지속적인 비용 절감



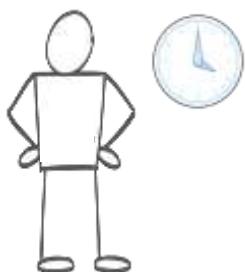
## 탄력적인 운영 및 확장

필요 용량에 대한 예측 불필요  
수요에 맞춘 유연한 확장



## 속도 및 민첩성

수 분 만에 인프라 구축 가능  
빠르게 변화에 대응



## 비즈니스에만 집중 가능

혁신을 위한 다양한 실험 가능  
불필요한 인프라 관리 업무 제거



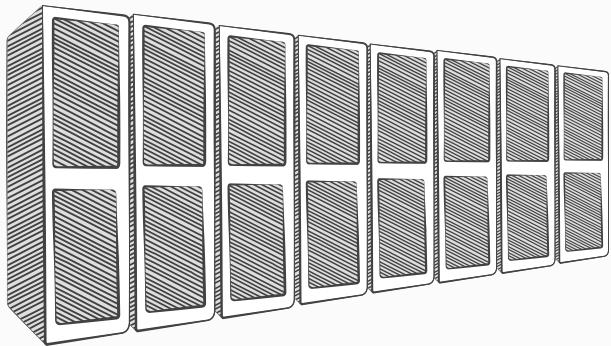
## 글로벌 확장

빠른 시간 내 글로벌 서비스  
구현 가능



# 왜 Cloud Computing 인가? – 1) 초기 선투자 비용 없음

On-premise



VS

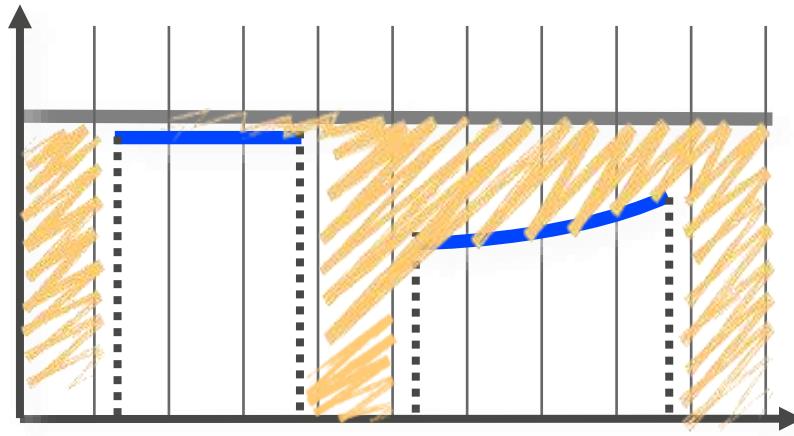
Cloud



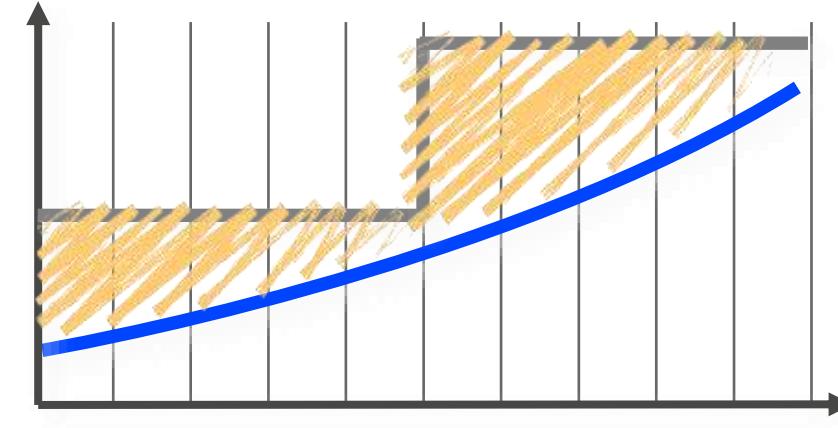
- ✓\$\$\$\$\$\$\$\$
- ✓ 많은 초기 자본 투자 비용 발생

- ✓ 초기 투자 비용: \$0
- ✓ 사용한 만큼만 지불

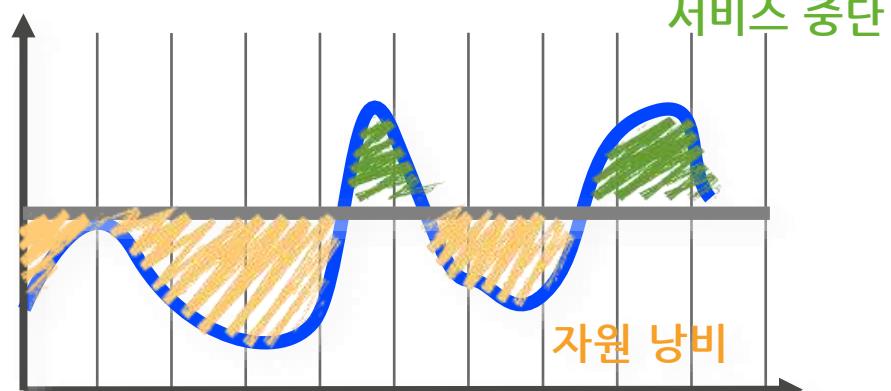
# 왜 Cloud Computing 인가? – 2) 운영 비용 절감



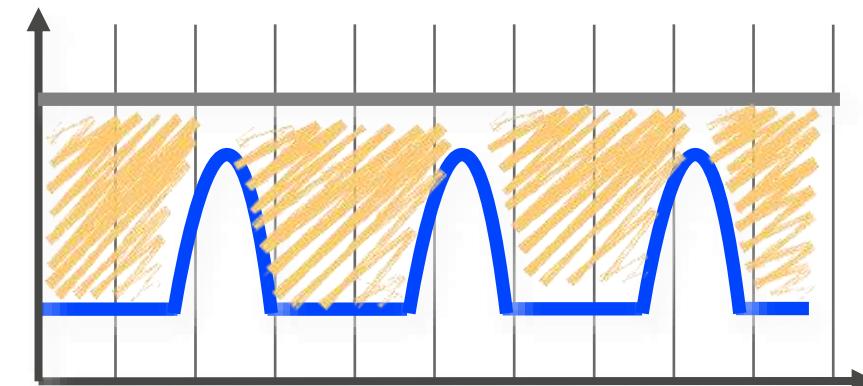
On and Off



Fast Growth



Variable peaks



Predictable peaks

# 왜 Cloud Computing 인가? – 2) 운영 비용 절감



전력 비용



항온항습 비용



상면 비용



운영 관리 비용



라이선스 비용

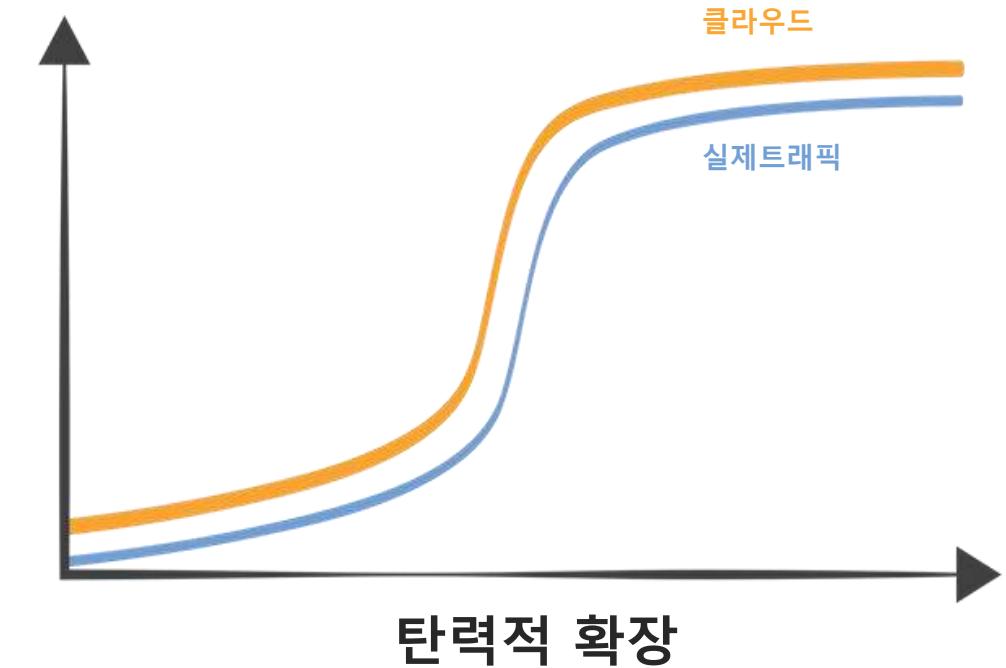
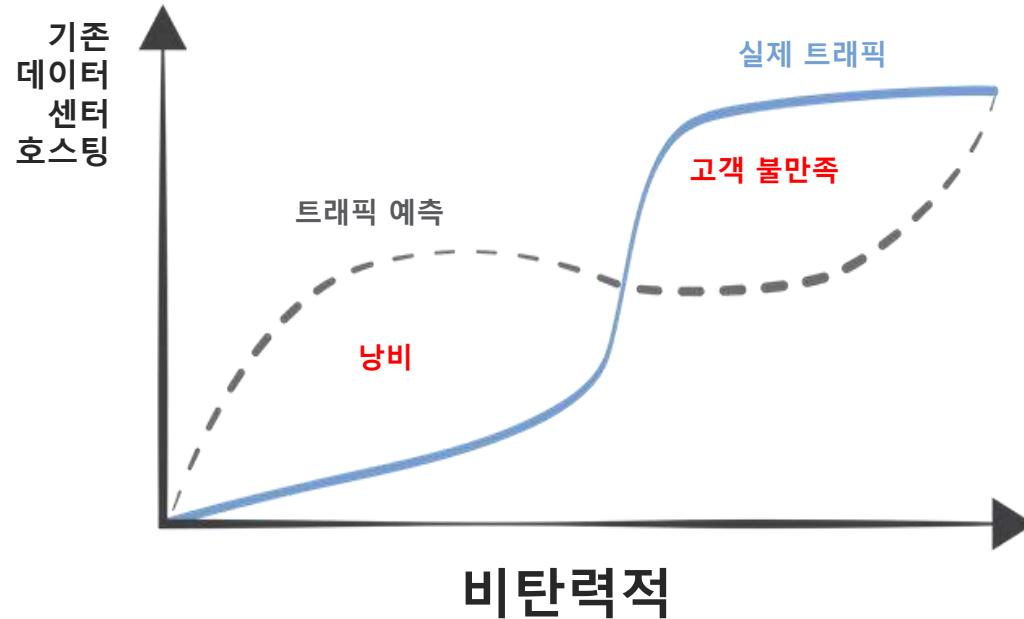


향후 증설 비용

“간과하기 쉬운 숨은 비용들 – on-premise / 데이터센터 이용 시”

# 왜 Cloud Computing 인가? – 3) 탄력적인 운영 및 확장

클라우드 컴퓨팅에서는 프로그램 코드로 필요한 자원을 자동 증설 및 감소 할 수 있어  
비용 효율적일 뿐만 아니라 최적의 성능 및 안정성을 제공 가능



# 왜 Cloud Computing 인가? – 4) 속도 및 민첩성

“수 주일 내 인프라 준비”



## On-Premises

혁신을 위한 시도가 자주 일어나지 않고

실패의 비용이 높음

혁신 속도가 느려짐

“수 분 내 인프라 준비”



혁신을 위한 시도를 많이 할 수 있고

실패의 비용이 낮음

많은 혁신이 가능

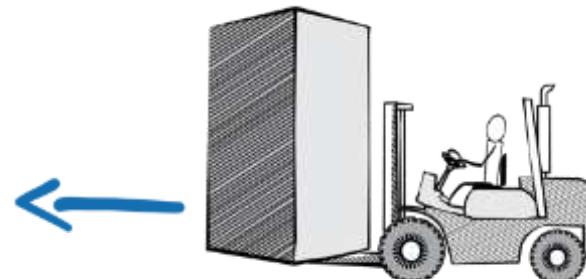
# 왜 Cloud Computing 인가? – 5) 비즈니스에만 집중 가능

클라우드 업체가 대신 관리

데이터 센터	네트워킹
전력 공급	서버 랙 관리
냉각/공조	서버 관리
케이블 연결	스토리지
보안 관리	시설 운영 등

신경쓸 필요가 없음

신규 하드웨어 구매 및 설치
신규 소프트웨어 설치 및 구성
데이터 센터 구축 및 업그레이드 등



“우리에게 전략적이지 않은 영역들,  
이를 테면 데이터 센터 운영과 같은 영역은  
클라우드에 맡기기로 결정하였습니다.”

- Mark Pincus, CEO



# 왜 Cloud Computing 인가? – 6) 글로벌 확장

전 세계 어디라도 수 분 내 확장하여 서비스 구축 가능



# 왜 Cloud Computing 인가? – 혁신 사례

- Global Unicorn Case



Learn from

“Instagram”

\$1 Bn in 2 Years by 12 People

- Korean Unicorn Cases



Online Shopping Mall

Tourism & Hotel Booking



Mobile Game



Food Delivery

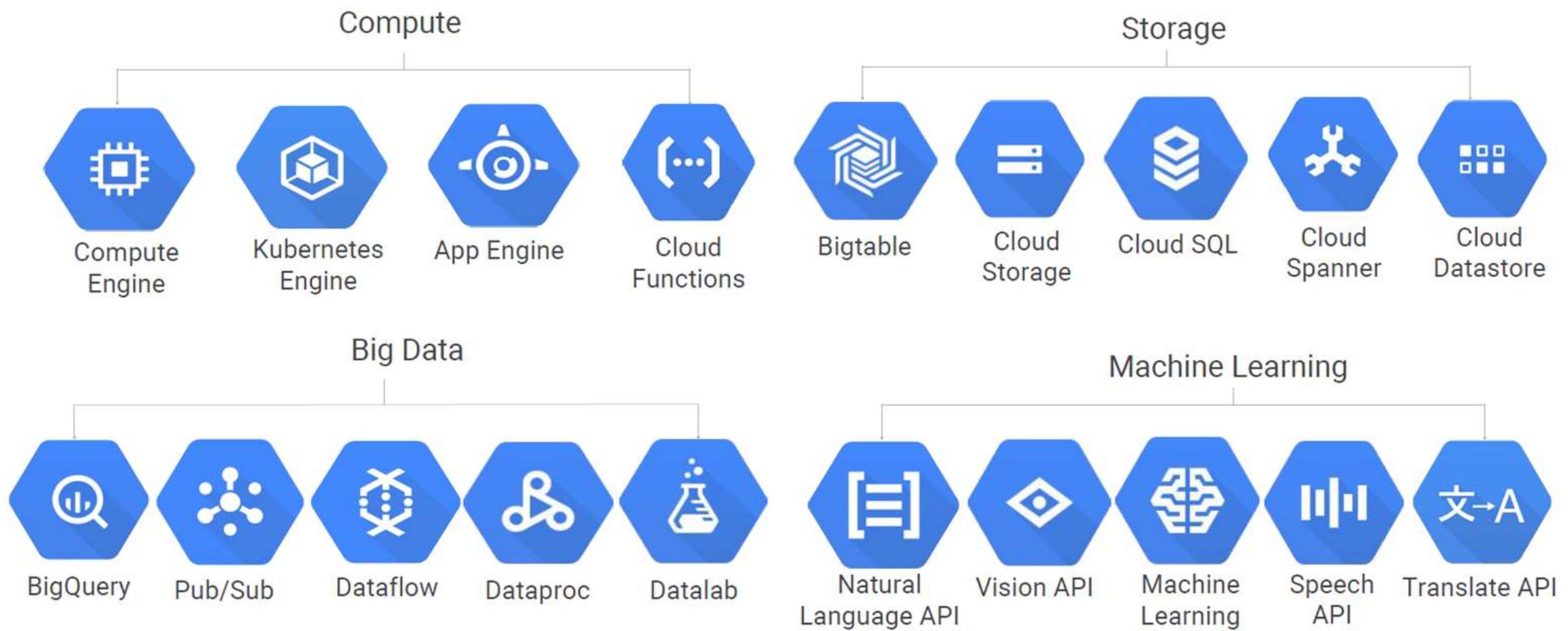


Real Estate Brokerage

# 클라우드 서비스 분류

<b>SaaS</b> Software as a Service					
<b>PaaS</b> Platform as a Service					
<b>IaaS</b> Infrastructure as a Service					

# 클라우드 자원과 서비스



# 글로벌 확장성 (Scalability)

- Amazon AWS
  - **22 Regions**
  - **69 Zones**
  - **245 Countries**



- Microsoft Azure
  - **56 Regions**
  - **100+ Zones**
  - **140 Countries**



- Google GCP
  - **20 Regions**
  - **61 Zones**
  - **200+ Countries**

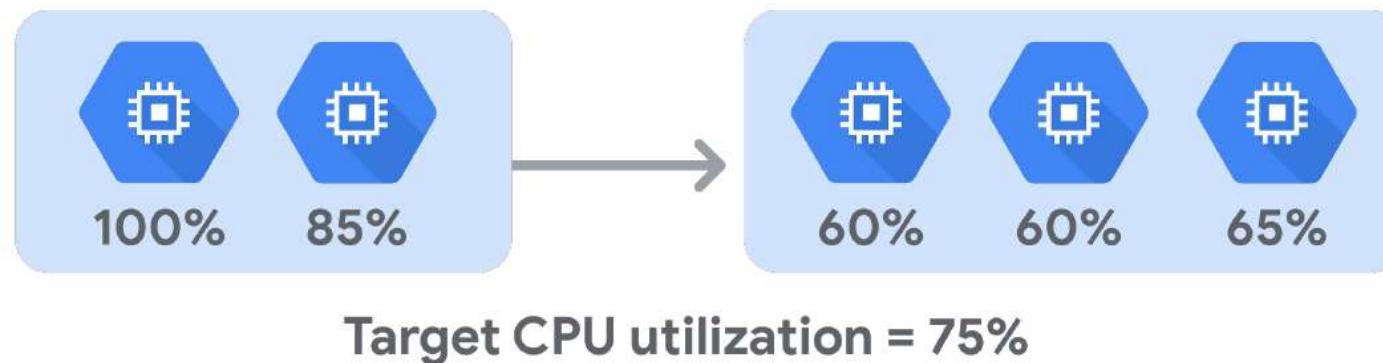


As of Feb 2020

<https://aws.amazon.com/ko/about-aws/global-infrastructure/?p=ngi&loc=0>  
<https://azure.microsoft.com/en-us/global-infrastructure/regions/>  
<https://cloud.google.com/about/locations/>

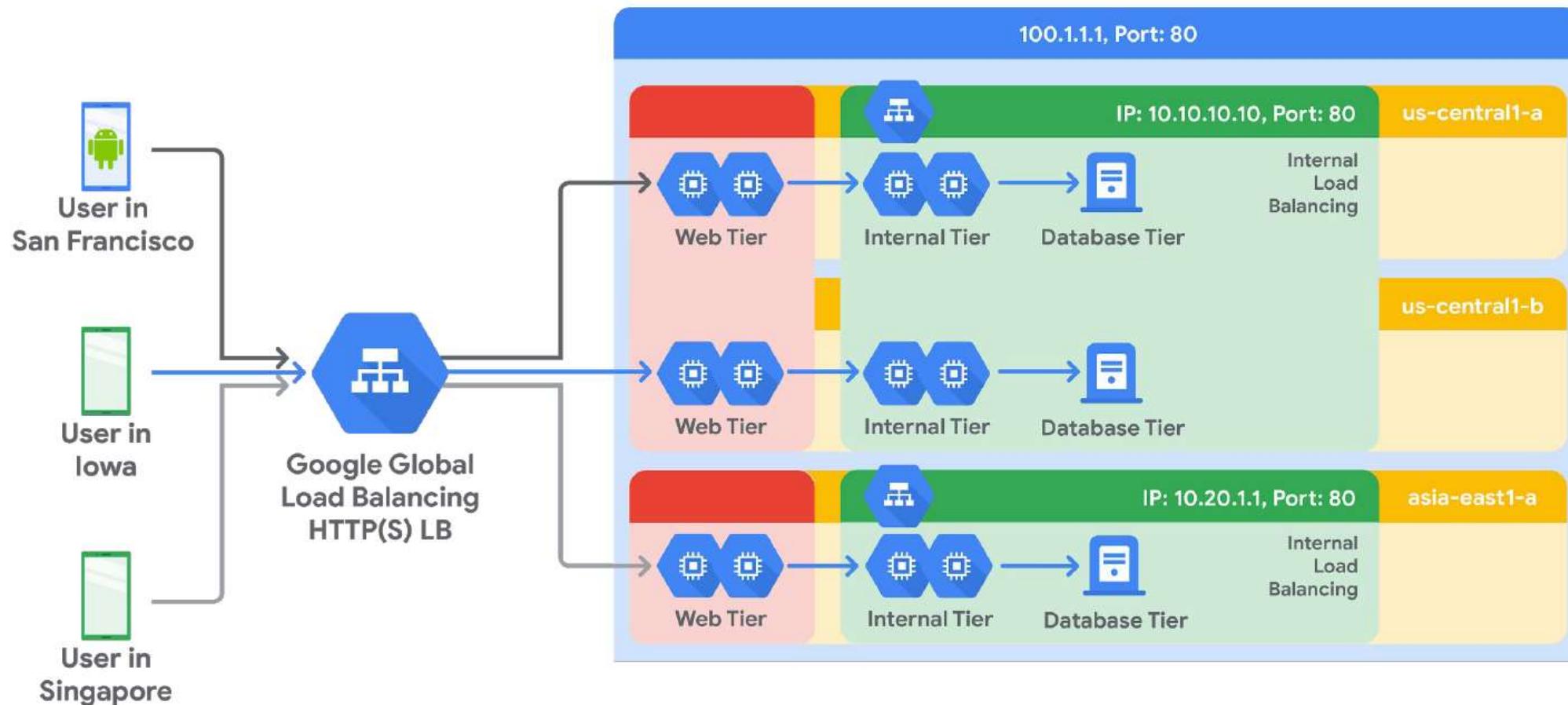
# 오토스케일링 (Auto-Scaling)

- Dynamically add/remove instances:
  - Increases in load
  - Decreases in load
- Autoscaling policy:
  - CPU utilization
  - Load balancing capacity
  - Monitoring metrics
  - Queue-based workload

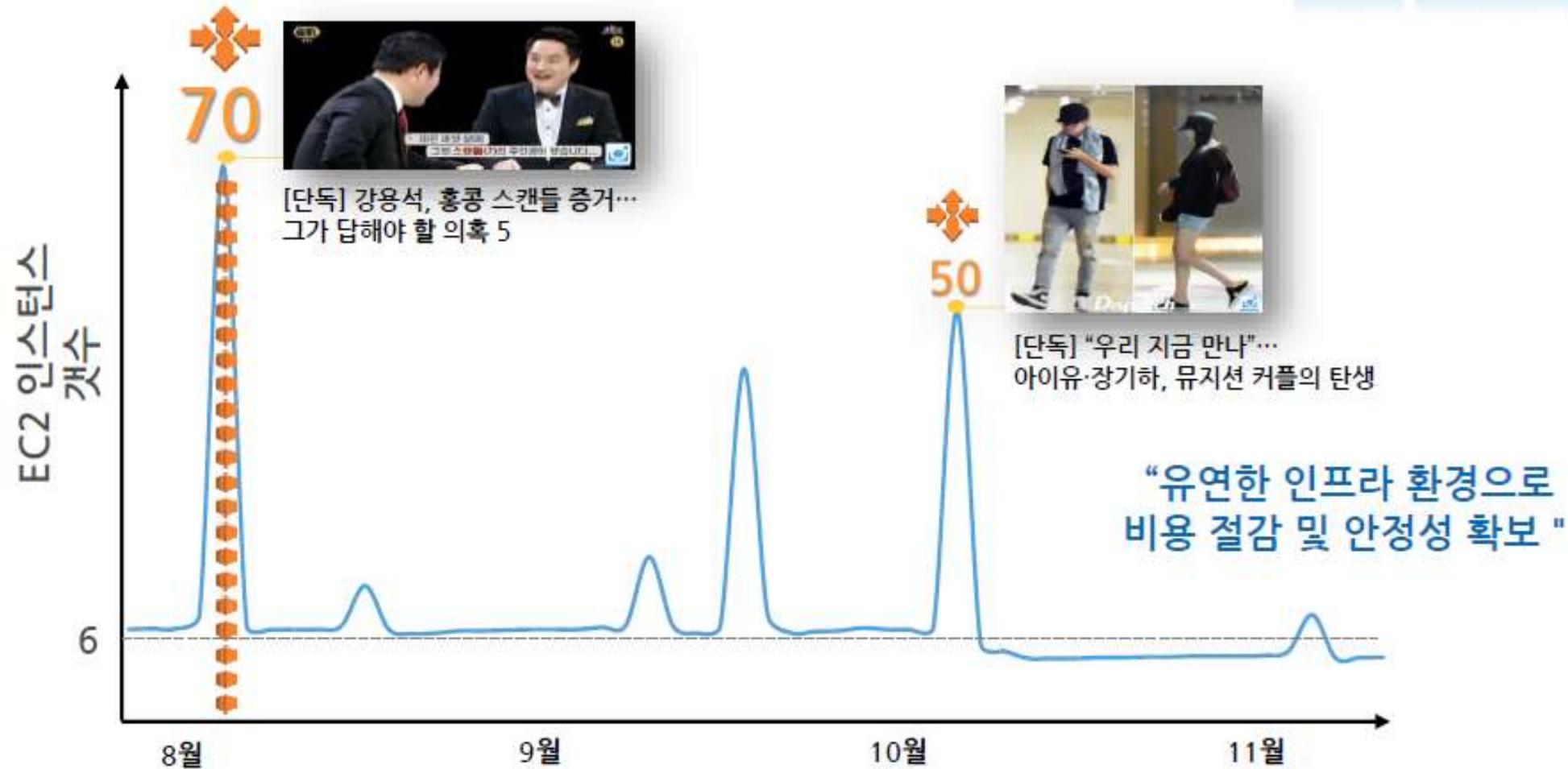


# 로드밸런싱 (Load-Balancing)

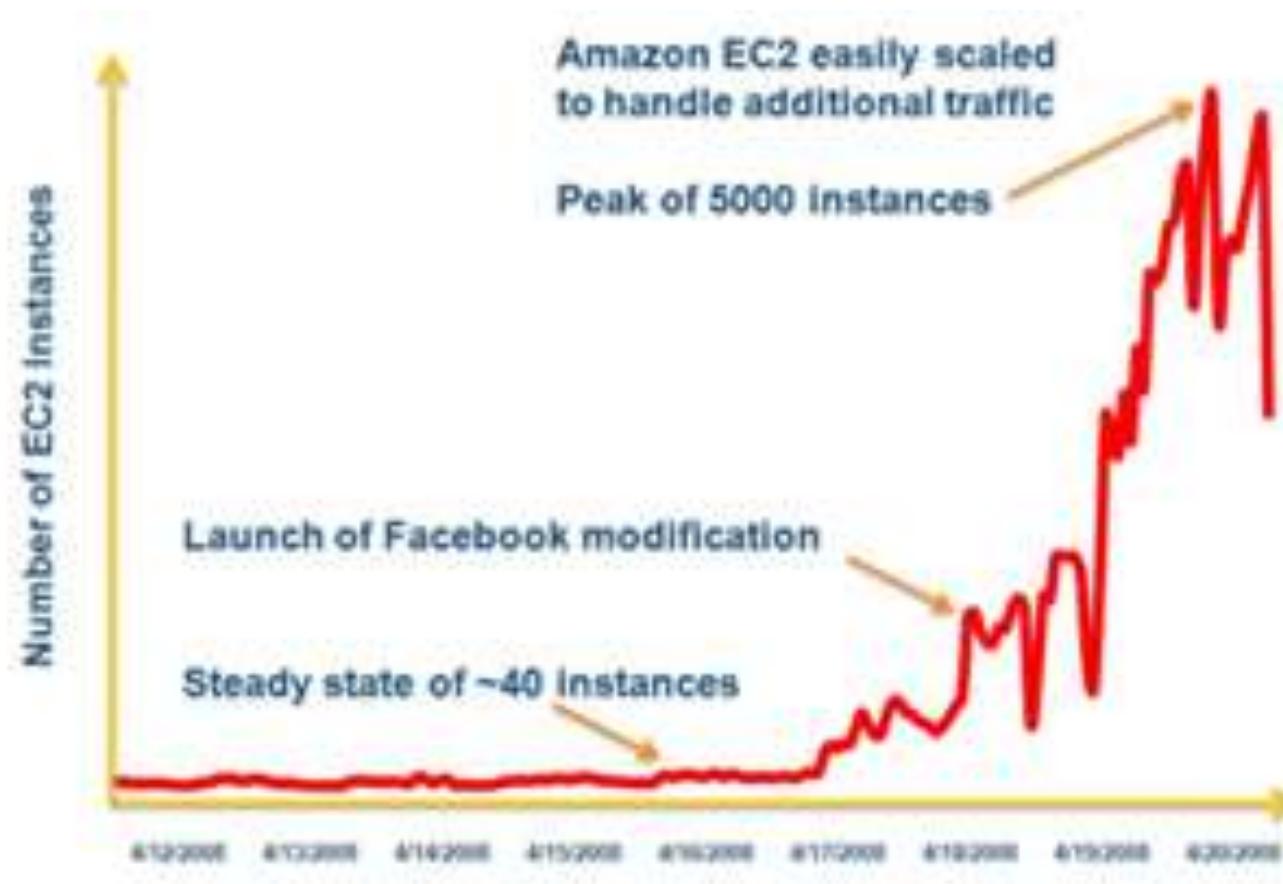
## 3-Tier Load Balancing Concept in Native Cloud Architecture



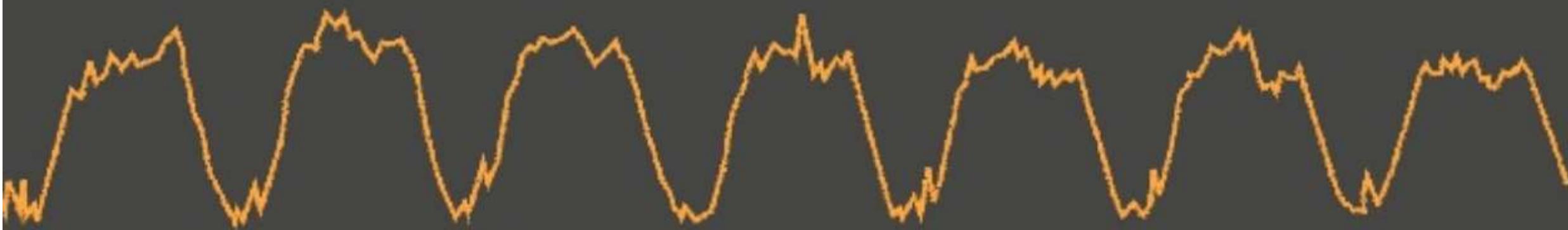
# Auto-Scaling 사례 - 디스패치



## 탄력적인 운영 확장 사례 - Animoto



## Typical weekly traffic to Amazon.com



Sunday

Monday

Tuesday

Wednesday

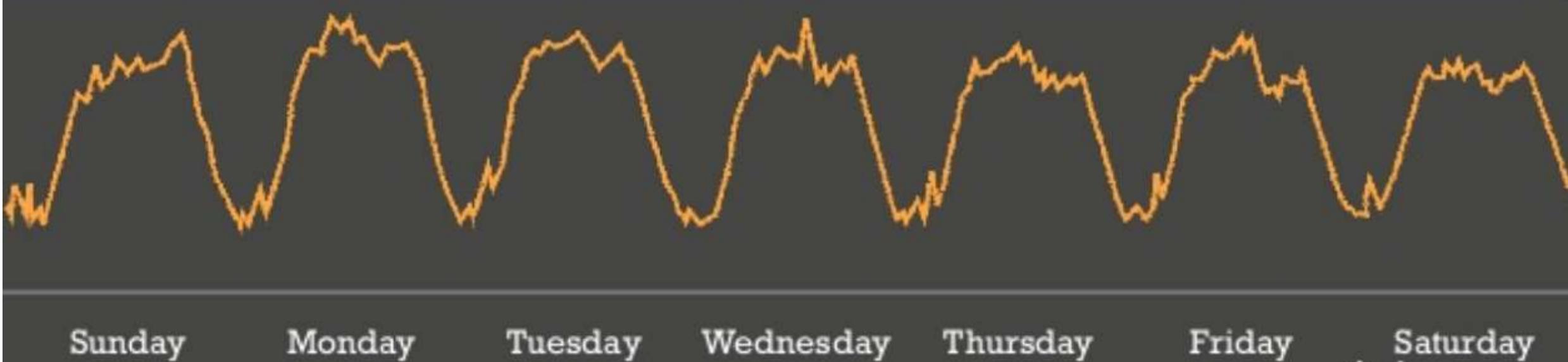
Thursday

Friday

Saturday

## Typical weekly traffic to Amazon.com

Provisioned capacity



Sunday

Monday

Tuesday

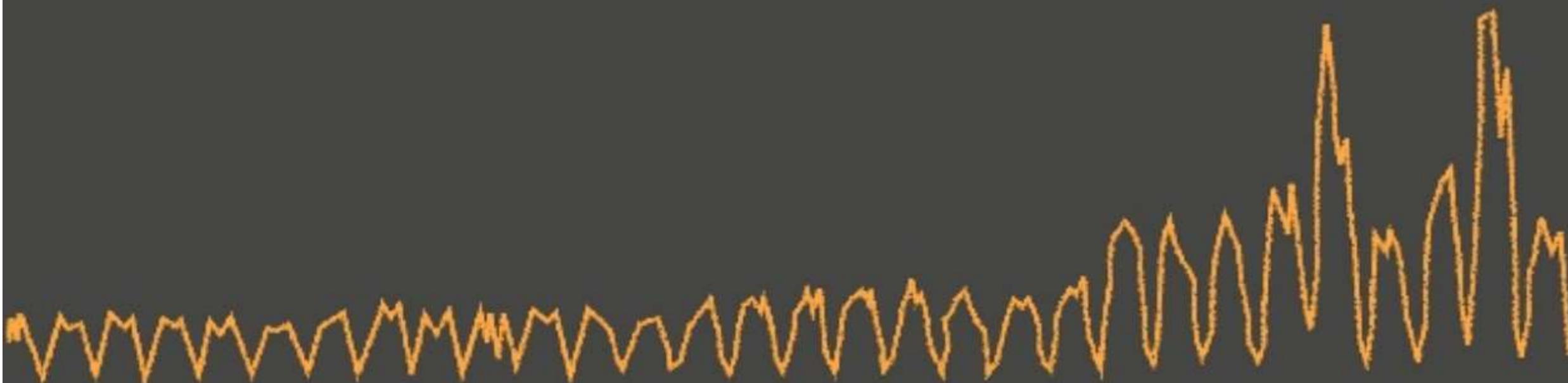
Wednesday

Thursday

Friday

Saturday

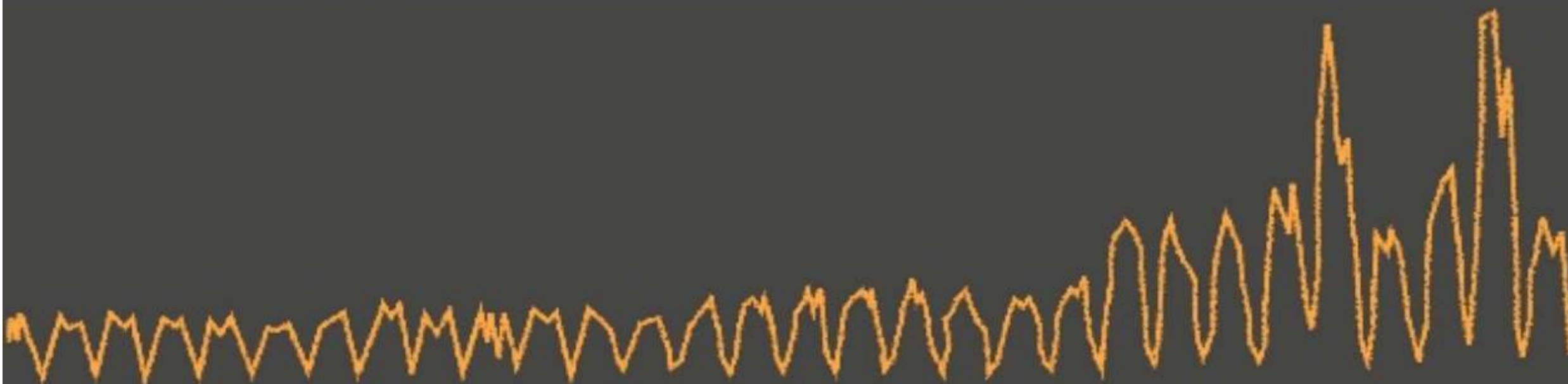
## November traffic to Amazon.com



November

## November traffic to Amazon.com

Provisioned capacity



November

## November traffic to Amazon.com

Provisioned capacity

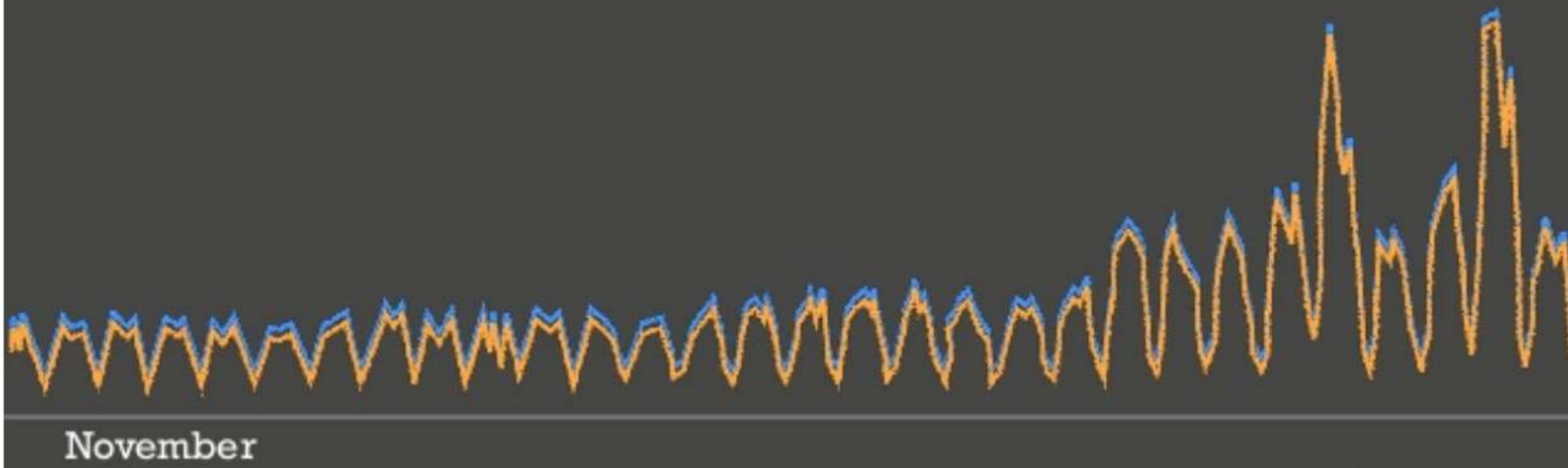
76%

November

24%



## November traffic to Amazon.com



# **Part III**

## **AMG 비교**

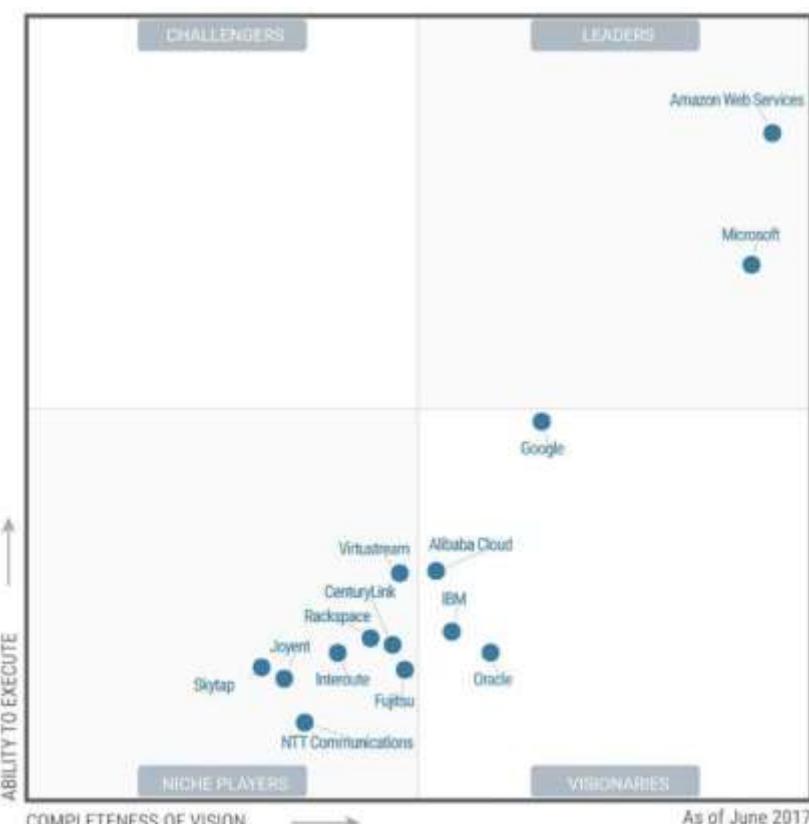
### **(Amazon AWS vs. Microsoft Azure vs. Google GCP)**

# 퍼블릭 클라우드 업체 비교 (IaaS)

2014년 7월



2017년 7월



2019년 7월



Gartner Magic Quadrant (IaaS)

# AMG Infrastructure 비교

- Amazon AWS
  - **22 Regions**
  - **69 Zones**
  - **245 Countries**



- Microsoft Azure
  - **56 Regions**
  - **100+ Zones**
  - **140 Countries**



- Google GCP
  - **20 Regions**
  - **61 Zones**
  - **200+ Countries**



As of Feb 2020

<https://aws.amazon.com/ko/about-aws/global-infrastructure/?p=ngi&loc=0>

<https://azure.microsoft.com/en-us/global-infrastructure/regions/>

<https://cloud.google.com/about/locations/>

# Compute Service 비교

- 전반적으로 모두 안정되고 훌륭한 성능이며 큰 차이가 없음
- 전반적으로 AWS > Azure > GCP 순으로 비용이 내려감
- AWS는 Spot Instance를 잘 활용해야 비용이 절감 가능하며, Azure는 기존 Windows Server와의 Hybrid를 잘 활용 해야하고, GCP는 알아서 깍아주는 지속사용할인이 장점이며, Kubernetes 서비스가 앞서 있음

Services	AWS	Azure	GCP
IaaS	Amazon Elastic Compute Cloud (EC2)	Virtual Machines	Google Compute Engine (GCE)
PaaS	AWS Elastic Beanstalk	App Service and Cloud Services	Google App Engine (GAE)
Containers	Amazon Elastic Compute Cloud Container Service (ECS)	Container Instances	Google Kubernetes Engine (GKE) & <a href="#">Cloud Run</a>
Kubernetes	Amazon Elastic Kubernetes Service (EKS)	Azure Kubernetes Service (AKS)	
Serverless Functions	AWS Lambda	Azure Functions	Google Cloud Functions
Special Cost Scheme	Reserved Instance (RI) <a href="#">Spot Instance</a>	Reserved VM Instance (RI) <a href="#">Azure Hybrid</a>	<a href="#">지속 사용 할인 (SUD)</a> 약정 사용 할인 (CUD) 선점형 VM 인스턴스

# Database Service 비교

- AWS 와 Oracle이 전반적으로 앞서 있으며, Azure는 전략적 방향성면에서 다소 부족하며, GCP는 제공하는 서비스 Coverage가 다소 부족하다고 평가됨



Forrest Wave DB-as-a-Service, 2Q 2019

<https://cloud.google.com/forrester-dbaas/?hl=ko>

# Database Service 비교

- 쉽게 쓰는 관리형 성능면에서는 Amazon Aurora (MySQL+PostgreSQL)가 뛰어나며,
- 기존 On-Prem 과 함께쓰는 비용면에서는 MS Azure SQL 이 가성비가 좋고,
- Data warehouse로 On-line Analytic Processing (OLAP) 사용 용도로는 BigQuery 가 앞도적임
- 특수한 용도로 Android 와 연계를 하기 쉬운 Firebase 나 글로벌 스케일로 확장이 가능한 관계형 DB 인 Cloud Spanner 가 있음

Services	AWS	Azure	GCP
RDBMS	RDS (Relational Database Service)	SQL Database	Cloud SQL
NoSQL: Key–Value	DynamoDB	Table Storage	Cloud Datastore / Cloud Bigtable
NoSQL: Indexed	Amazon SimpleDB	Azure Cosmos DB	Cloud Datastore
Data Warehouse	Amazon Redshift	Azure SQL Data Warehouse	BigQuery
Remark	Amazon Aurora	Azure Hybrid	Google Firebase / Google Cloud Spanner

# Storage Service 비교

- 전반적으로 모두 안정되고 훌륭한 성능이며 큰 차이가 없음
- 전반적으로 AWS > Azure > GCP 순으로 비용이 내려감

Services	AWS	Azure	GCP
<b>Object Storage</b>	S3 (Simple Storage Service)	Disk Storage	Cloud Storage
<b>Block Storage</b>	EBS (Elastic Block Store)	Blob Storage	Persistent Disks
<b>Cold Storage</b>	Glacier	Archive Blob Storage	Cloud Storage Nearline/Coldline
<b>File Storage</b>	Amazon Elastic File System	Azure File Storage	ZFS/Avere

# Network Service 비교

- 전반적으로 모두 안정되고 훌륭한 성능이며 큰 차이가 없음
- 전반적으로 AWS > Azure > GCP 순으로 비용이 내려감

Services	AWS	Azure	GCP
<b>Virtual Network</b>	Amazon Virtual Private Cloud (VPC)	Virtual Networks (VNets)	Virtual Private Cloud
<b>Elastic Load Balancer</b>	Elastic Load Balancer	Load Balancer	Google Cloud Load Balancing
<b>Peering</b>	Direct Connect	ExpressRoute	Google Cloud Interconnect
<b>DNS</b>	Amazon Route 53	Azure DNS	Google Cloud DNS

# 전반적인 서비스 지원 여부 비교

<http://comparecloud.in/>라는 사이트에서 각 클라우드별로 어떤 서비스를 지원하는지 비교가 가능함

The screenshot shows a comparison table for various cloud services across six providers: Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform, IBM Cloud, Oracle, and Alibaba Cloud. The table is organized by service category (Compute, Storage, Networking, etc.) and provider.

		Category	Service	AWS	Azure	Google Cloud Platform	IBM Cloud	Oracle	Alibaba Cloud
Compute	Shared Web hosting				Azure shared App Services		Web hosting services		
Compute	Virtual Server	Amazon EC2		Azure Virtual Machine	Compute Engine	Classic Virtual Server	Compute		Alibaba ECS
Compute	Bare Metal Server	Amazon EC2 Bare Metal Instance (Preview)		Azure Bare Metal Servers (Large Instance Only for SAP Hana)		Bare Metal Servers	Bare Metal Servers		ECS Bare Metal Instance
Compute	Virtual Dedicated Host	Amazon EC2 Dedicated Hosts		Azure Dedicated Host	Sole Tenant Node (Beta)	Dedicated Virtual Servers Infrastructure (VSI)	Dedicated Compute Classic		ECS Dedicated Host
Compute	Container Registration Service	Amazon EC2 Container Registry		Azure Container Registry	Container Registry	IBM Cloud Container Registry	Oracle Cloud Infrastructure Registry		Container Registry
Compute	Container Management Service	Amazon EC2 Container Service	Amazon Elastic Container Service for Kubernetes (EKS)	Azure Kubernetes Service (AKS)	Azure Container Instances	IBM Cloud Kubernetes Service	Container Engine for Kubernetes (CKE)	Container Service	Container Service for Kubernetes
Compute	Micro Services App Development Platform	AWS Lambda		Azure Service Fabric	Google Cloud Functions	IBM Cloud Functions	Oracle Functions		Function Compute
Compute	Virtual Private Servers	Amazon Lightsail		Azure App Service Environment		Classic Virtual Server			Simple Application Server
Compute	Auto Scaling	Auto Scaling		Azure AutoScale	Auto Scaler	Auto Scaling	Auto Scaling		ECS Auto Scaling

<http://comparecloud.in/>

# MS Azure의 AWS 와 비교 사이트

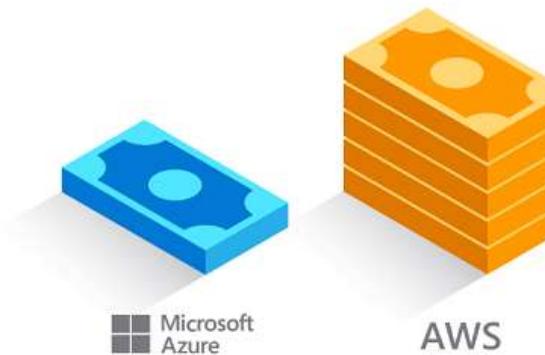
AWS보다 후발주자인 Azure의 경우 AWS vs. Azure 비교를 상세하게 해 놓았음

The screenshot shows the Microsoft Azure homepage with a dark theme. At the top, there's a navigation bar with links like '개요' (Overview), '솔루션' (Solutions), '제품' (Products), '설명서' (Documentation), '가격' (Pricing), '교육' (Education), 'Marketplace', '파트너', '지원', '블로그', and '기타'. Below the navigation, there's a breadcrumb trail: '홈 / 개요 / Azure와 AWS'. The main title 'Azure와 AWS' is displayed prominently. A subtext below it reads: '전 세계 조직이 기업 및 하이브리드 인프라를 위한 가장 신뢰할 수 있는 클라우드로 AWS(Amazon Web Services)보다 Microsoft Azure를 인정합니다.' A green button labeled '체험 계정 만들기 >' is visible. The overall layout is clean and professional.

## Azure를 통해 적은 요금 지불

AWS는 Windows Server 및 SQL Server용  
Azure보다 5배 더 비쌉니다.

[비교: Azure 대 AWS 가격에 대해 자세히 알아보기 >](#)



<https://azure.microsoft.com/ko-kr/overview/azure-vs-aws/>

<https://docs.microsoft.com/ko-kr/azure/architecture/aws-professional/services>

# 구글 클라우드의 타 클라우드 전문가 지원 사이트

후발주자인 구글의 경우 타 클라우드 플랫폼별 전문가를 위한 별도의 설명이 잘 되어 있음

홈 > 문서



## 플랫폼 비교

### AWS 전문가를 위한 Google Cloud Platform

Google Cloud Platform과 AWS의 제품 및 기능을 비교합니다.

### Azure 전문가를 위한 Google Cloud Platform

Google Cloud Platform과 Microsoft Azure의 제품 및 기능을 비교합니다.

### 데이터 센터 전문가를 위한 Google Cloud Platform

Google Cloud Platform과 기존 데이터 센터 및 코로케이션 시설을 비교합니다.

### OpenStack 사용자를 위한 Google Cloud Platform

Google Cloud Platform의 기초적인 사항과 OpenStack 통합 또는 이전 방법에 대해 알아봅니다.

<https://cloud.google.com/docs/compare/?authuser=1&hl=ko>

# AMG 비교 요약

## Amazon AWS

- 국내 Infra가 많아서 빠름
- 국내 사용자 저변이 넓음
- 기술인력 구하기가 쉬움
- 가장 폭넓은 포트폴리오
- 컴플라이언스 국내 인증 다수
- 가격이 다소 비싼편
- 조직관리가 힘듦
- 자원의 가시성이 좋지 않음
- On-Prem과 호환이 잘 안됨

## MS Azure

- 기존 윈도우 사용자가 접근 용이
- 기존 License 이용시 저렴
- 조직관리에서 Active Directory 가 사실상의 표준
- 기술인력 구하기가 쉽지 않음
- 기존 독점에 대한 반감
- 윈도우 기반 vs 리눅스 기반 2가지 체계 운영을 하는 부담 (기술 인력 2종 필요 및 기존 제품들이 비쌈)

## Google GCP

- 사실상 가장 저렴
- 기존 구글 서비스와 연계 (G-Suite, Google Analytics)
- 데이터 분석 및 인공지능 성능
- 오픈소스 생태계 호환
- Hybrid / 멀티 운영에 호의적
- 안드로이드와 호환성 높음
- 기술인력 구하기가 매우 어려움
- 국내 오픈이 늦어서 컴플라이언스 국내 인증 적음
- 한글화 늦음

## Module 2

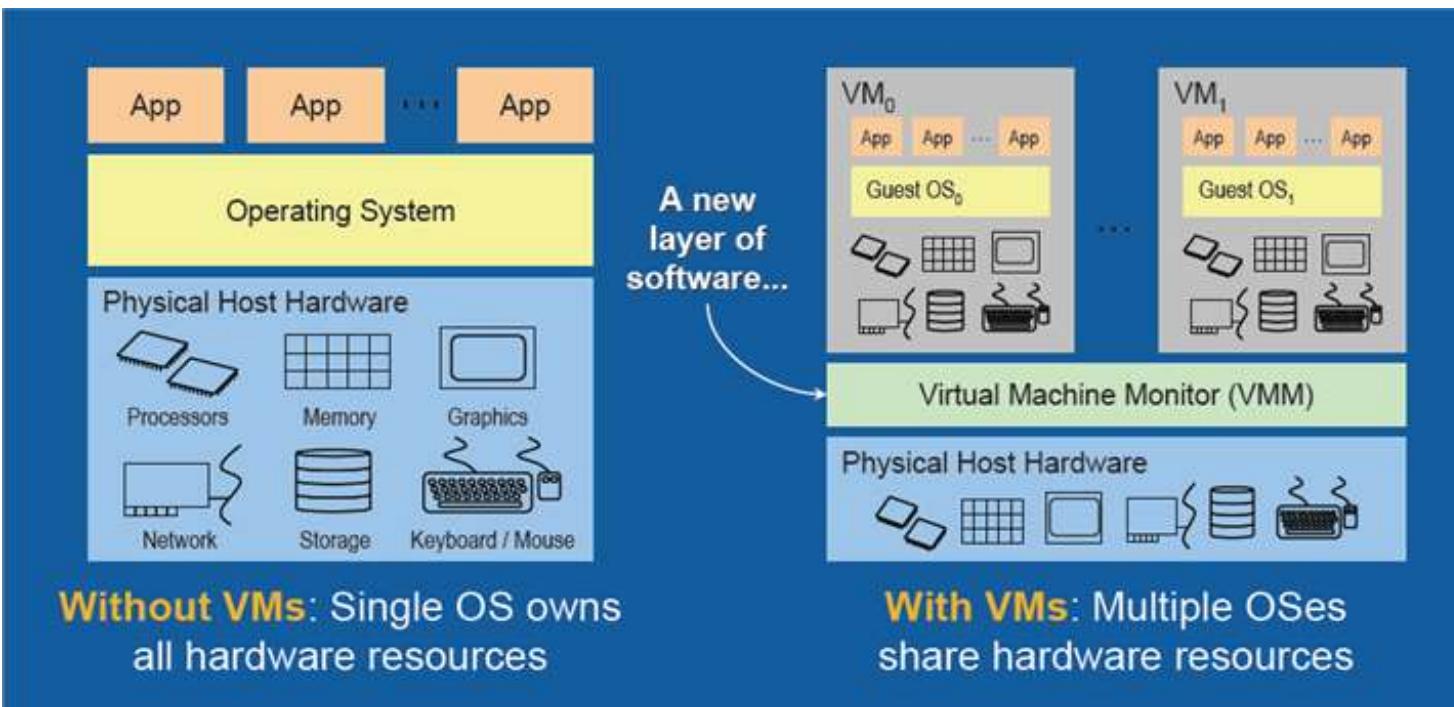
# 가상화 / 도커 컨테이너 / 쿠버네티스 핵심개념과 실습

# Part I

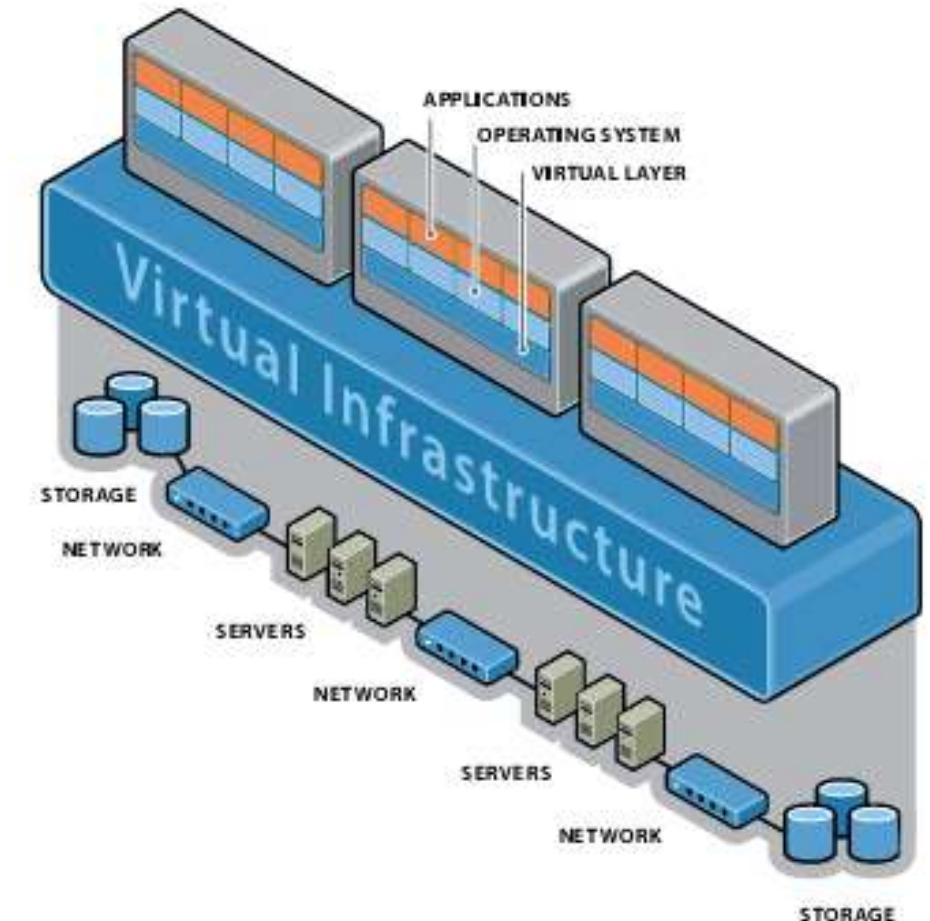
## 가상화 (Virtualization)

# 가상화란 ?

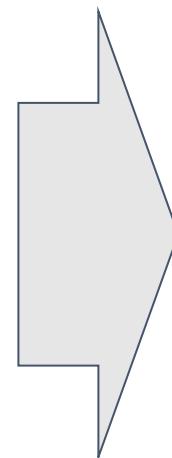
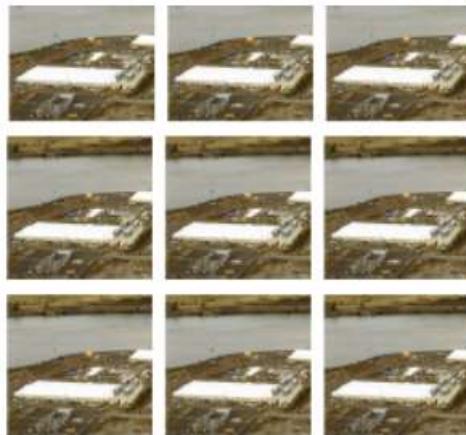
Virtualization means the creation of a virtual version of computing resources such as virtual computer hardware platforms, storage devices, computer network resources and etc.



Intel, Capitalhead and Wikipedia



# 가상화의 장점



**1. Reduce the Complexity**  
*to simplify operations and maintenance*



**2. Dramatically Lower Costs**  
*to redirect investment into value-add opportunities*



**3. Enable Flexible, Agile IT Service Delivery**  
*to meet and anticipate the needs of the business*



# 가상화된 서비스 예 : AWS



컴퓨팅



스토리지 및  
콘텐츠 전송



데이터베이스



네트워킹



관리 도구



보안 및  
자격 증명



분석



애플리케이션  
서비스



엔터프라이즈  
애플리케이션



인공 지능

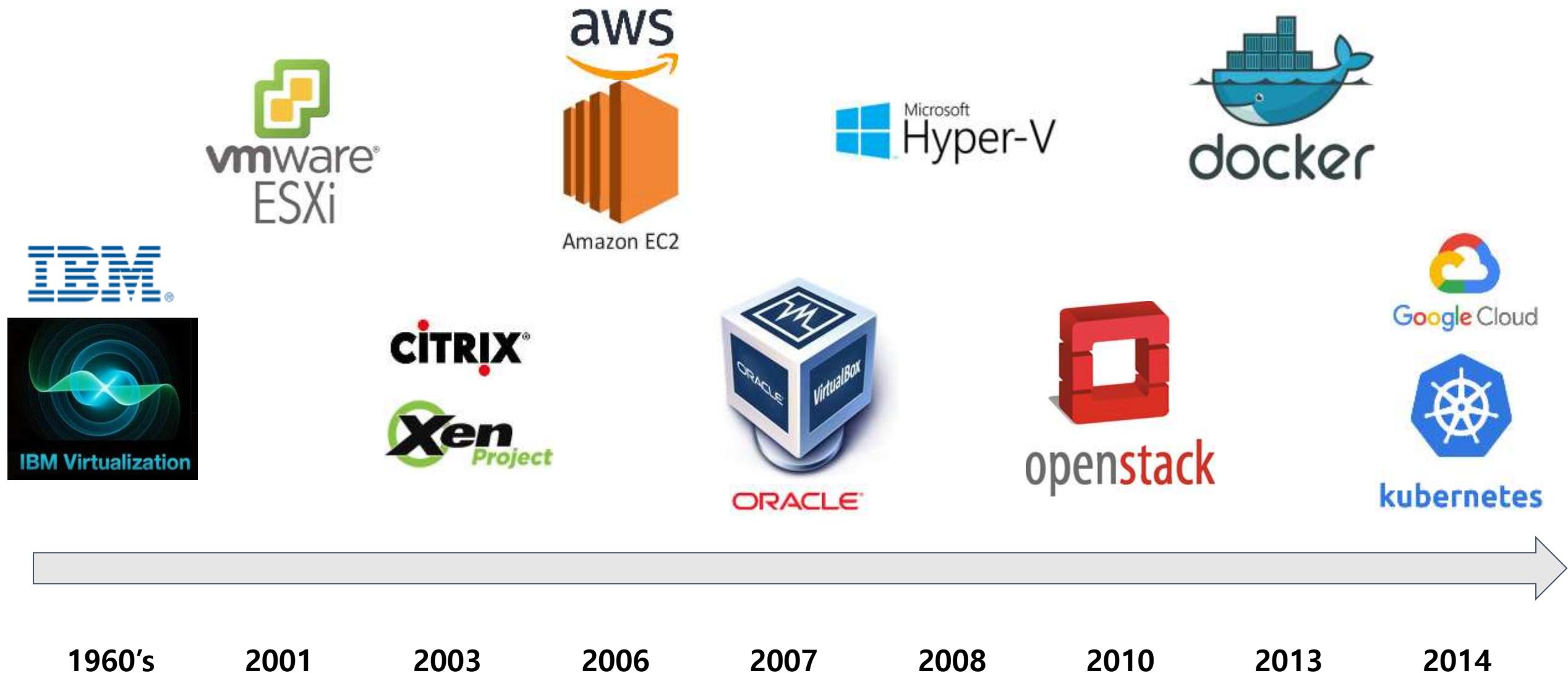


IoT



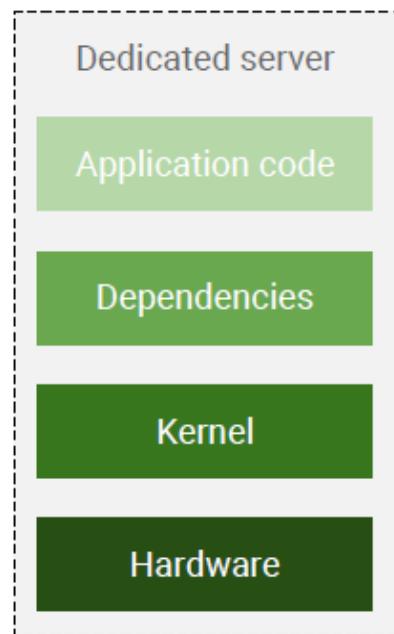
AWS 개발자  
도구

# 가상화의 역사

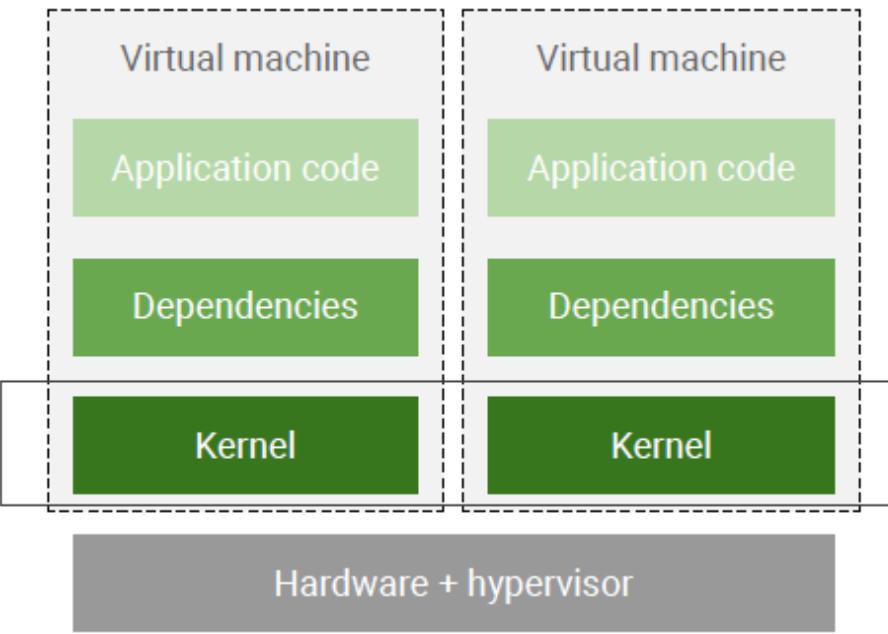


# 가상화 기술의 발전

## 호스트 가상화



## 하이퍼바이저 가상화

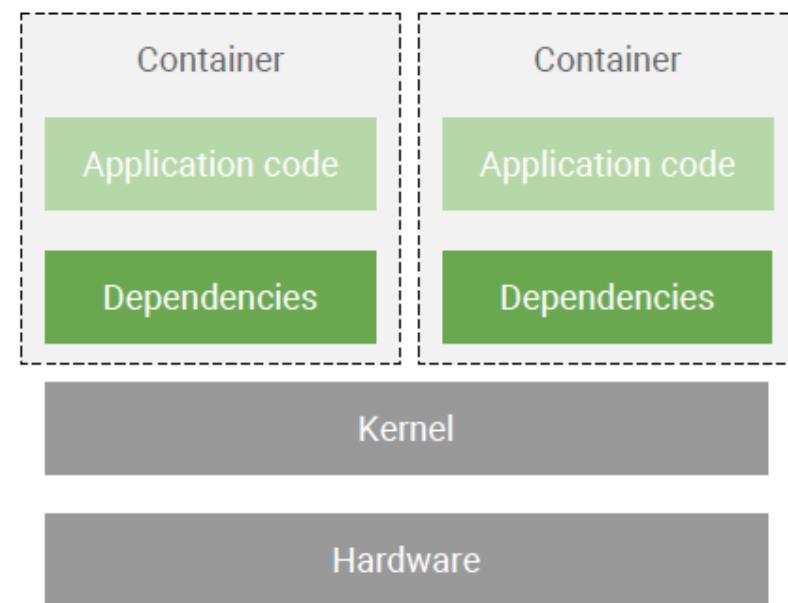


Deployment ~months  
Low utilization  
Not portable

Deployment ~days (mins)  
Hypervisor-specific  
Low isolation; tied to OS

Deployment ~days (mins)  
Hypervisor-specific  
**Redundant OS**

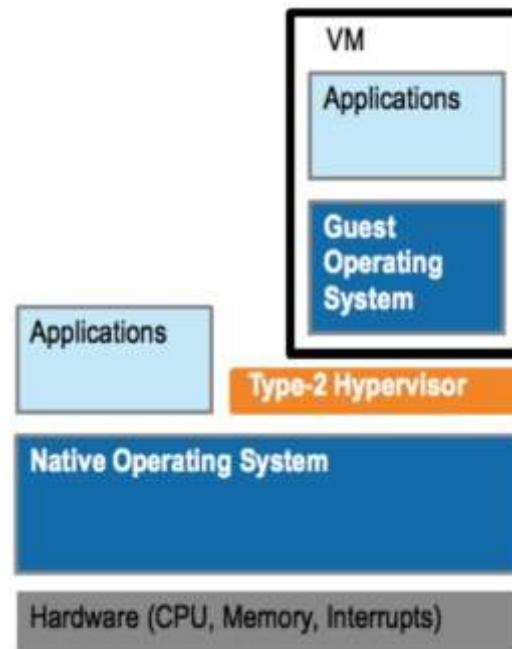
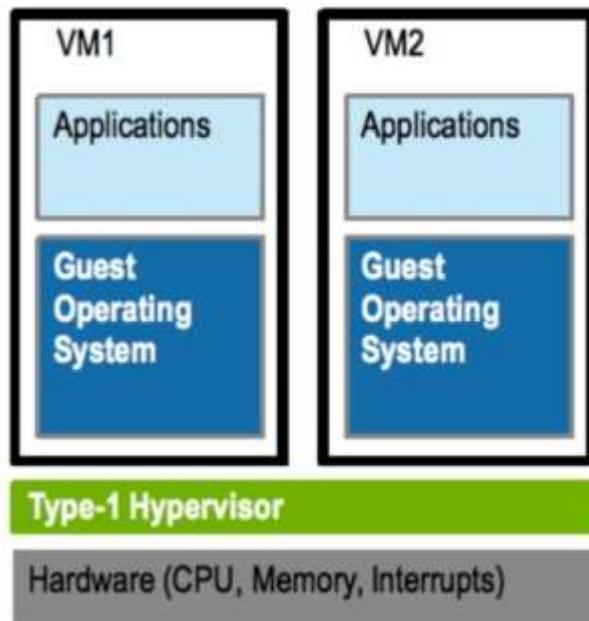
## 컨테이너 가상화



**Deployment ~mins (sec)**  
Portable  
Very efficient

# 하이퍼바이저(Hypervisor) 가상화의 종류

- Classification by Architecture Types

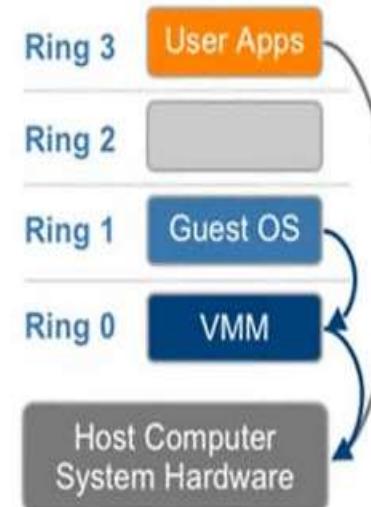


Type 1: Bare Metal Architecture  
Virtual Machine Monitor

Type 2: Hosted Architecture

- Sub-Classification of Type 1 Virtualization

## Full Virtualization



## Paravirtualization

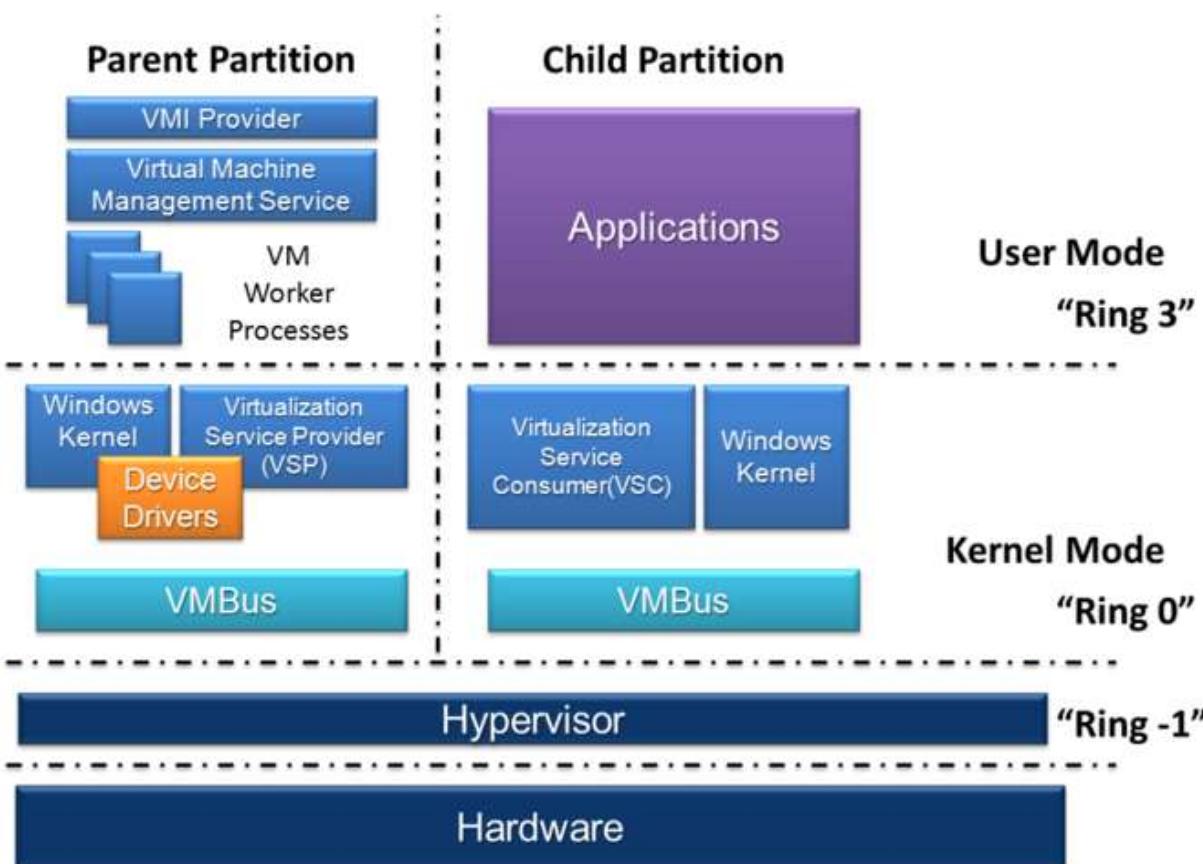


## Hardware Assisted

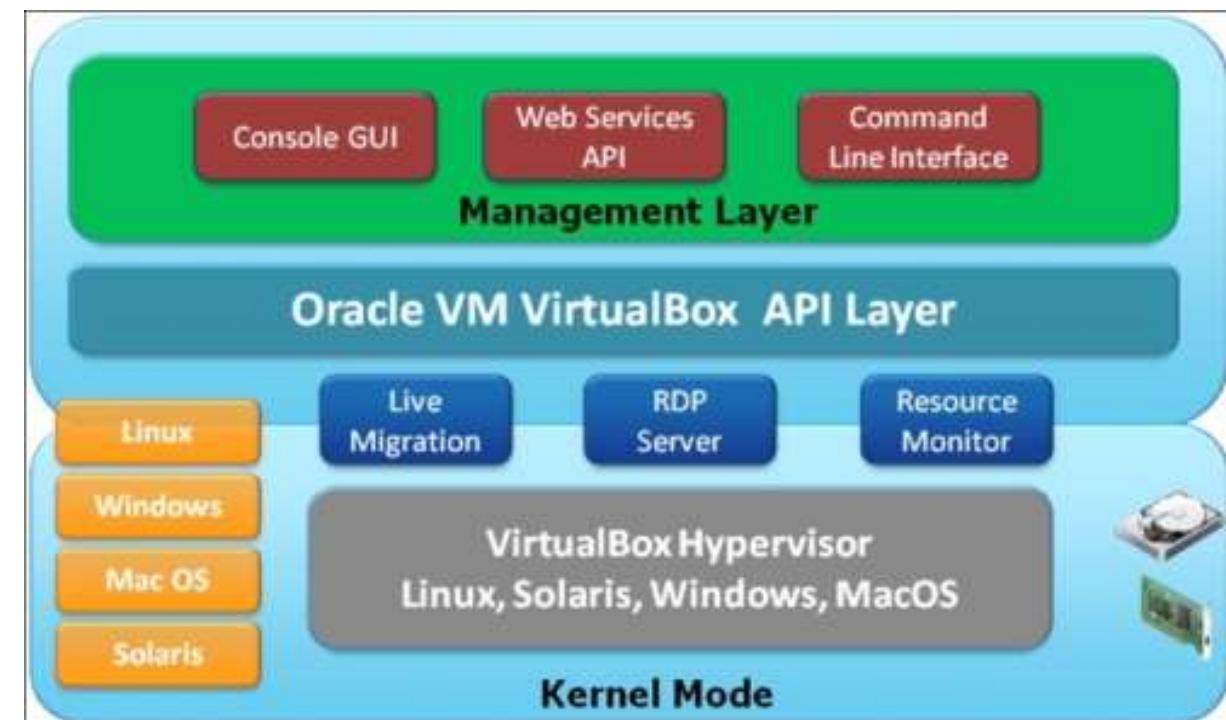


# 하이퍼바이저(Hypervisor) 가상화 사례

- MS Hyper-V Architecture



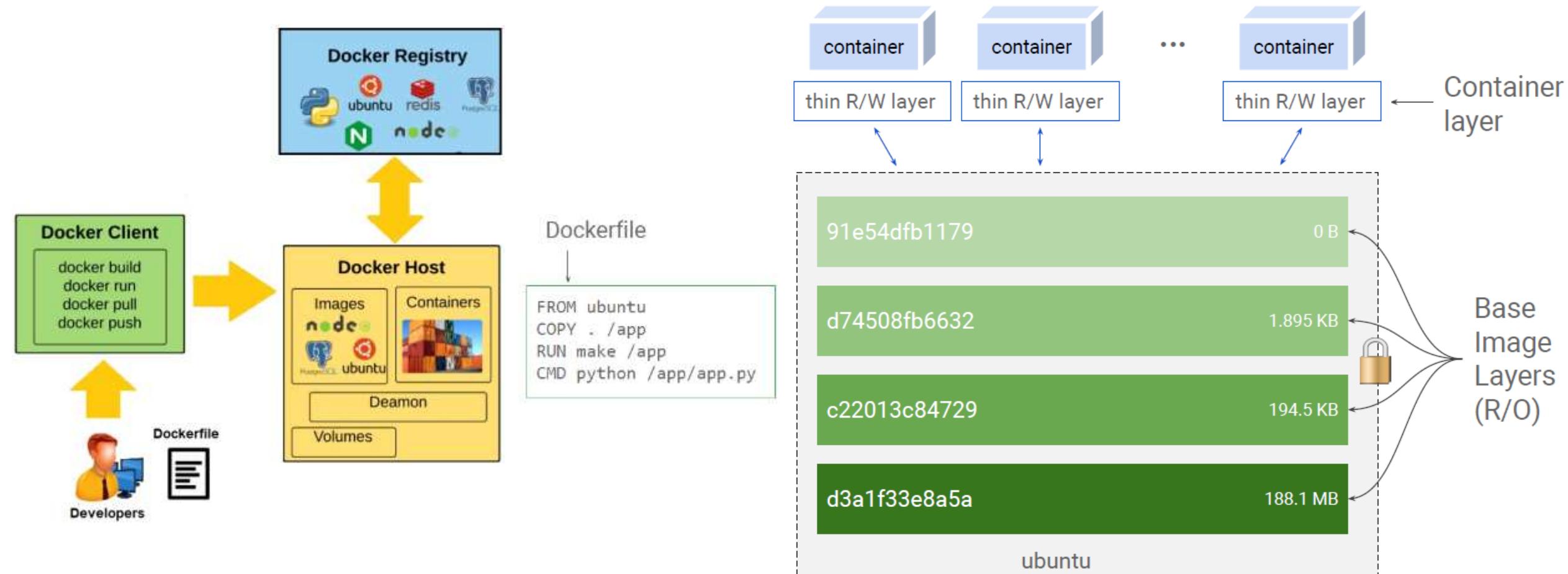
- Oracle Virtualbox Architecture



# 도커 컨테이너 이미지 (Docker Container)

- 컨테이너 기술 중 사실상의 표준 (De facto standard)은 Docker
- AWS 기반 컨테이너에 대한 자세한 내용은 링크 참조

[https://aws.amazon.com/ko/containers/?nc1=f\\_cc](https://aws.amazon.com/ko/containers/?nc1=f_cc)



Reorganized from DZone, Google & Coursera

# 도커 컨테이너 사용 예

- Install Docker Engine - Community (Ubuntu Case)
  - <https://docs.docker.com/install/linux/docker-ce/ubuntu/>

```
$ sudo apt-get update  
$ sudo apt-get install docker-ce docker-ce-cli containerd.io  
$ sudo docker run hello-world
```

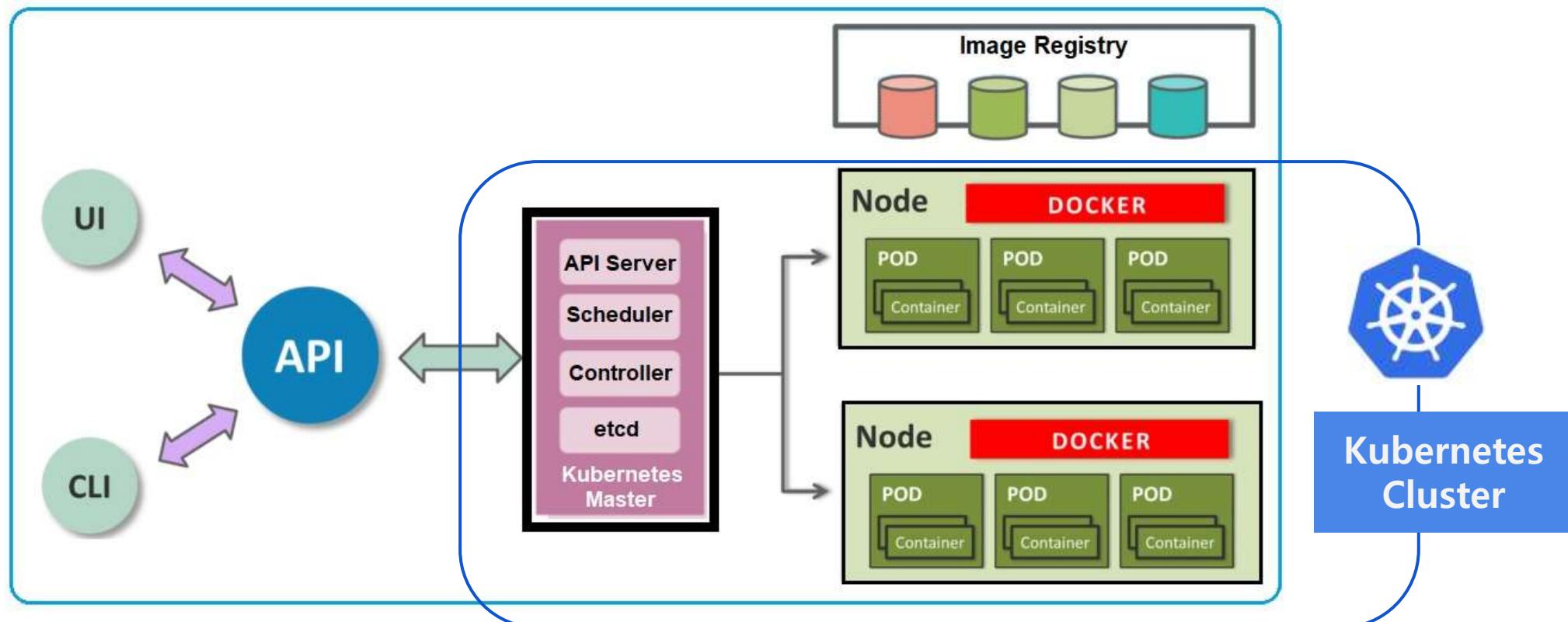
- Build & Run a Docker Image
  - <https://docs.docker.com/get-started/>

```
$ docker build --tag=imagename  
$ docker image ls  
$ docker push username/repository:tag  
$ docker run -p 4000:80 username/repository:tag
```

# 쿠버네티스 (Kubernetes, k8s)

- 컨테이너 오케스트레이션(Orchestration) 기술 중 사실상 표준
- 완전관리형의 Kubernetes 서비스

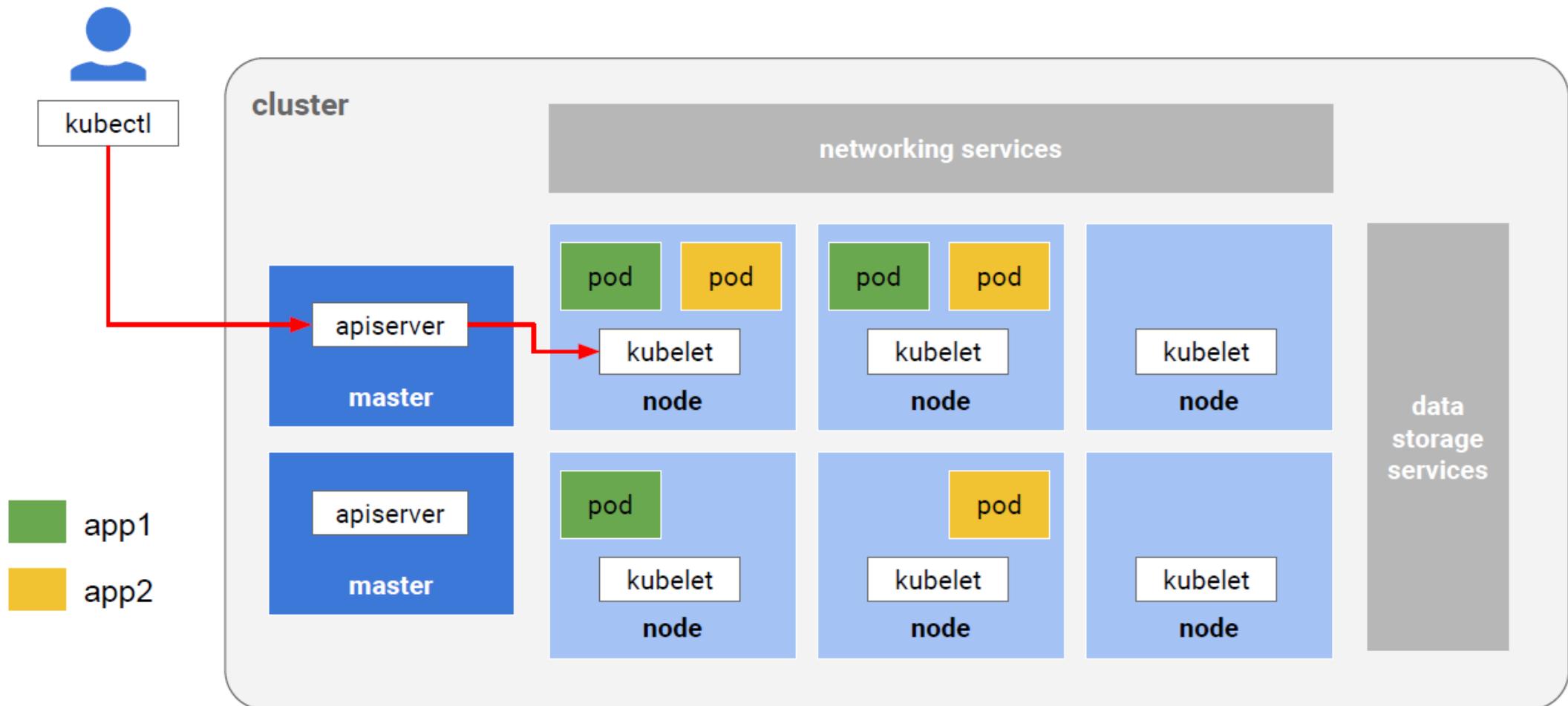
<https://aws.amazon.com/ko/eks/>



Reorganized from DZone (<https://dzone.com/articles/docker-containers-and-kubernetes-an-architectural>)

# 쿠버네티스 아키텍처

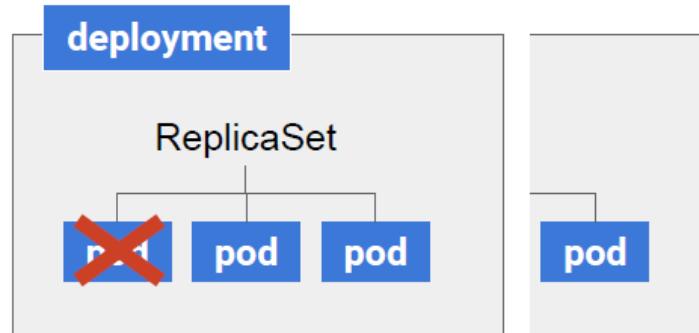
Kubernetes provide a control tool 'kubectl' to manage workloads inside the cluster.



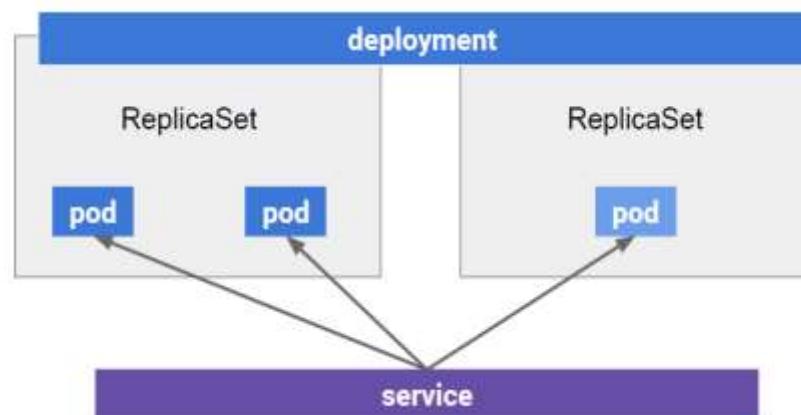
# 클라우드 쿠버네티스 엔진

Kubernetes supports many features to minimize downtime such as auto-scaling, rolling updates and blue/green deployment.

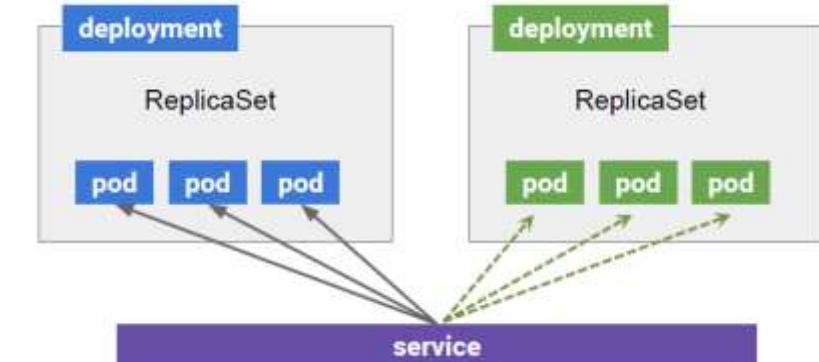
Auto-Scaling



Rolling Updates

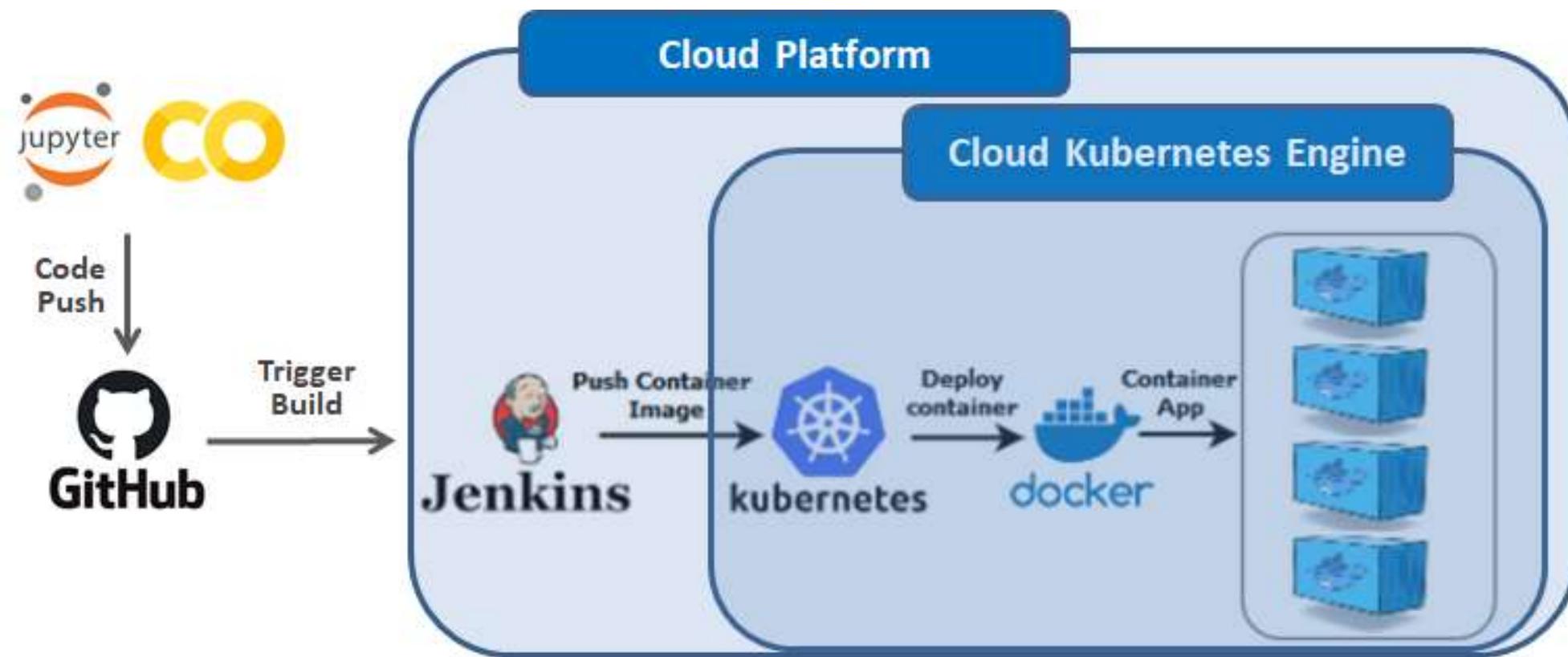


Blue Green Deployment



# 하이브리드/멀티클라우드 운영 – CI/CD

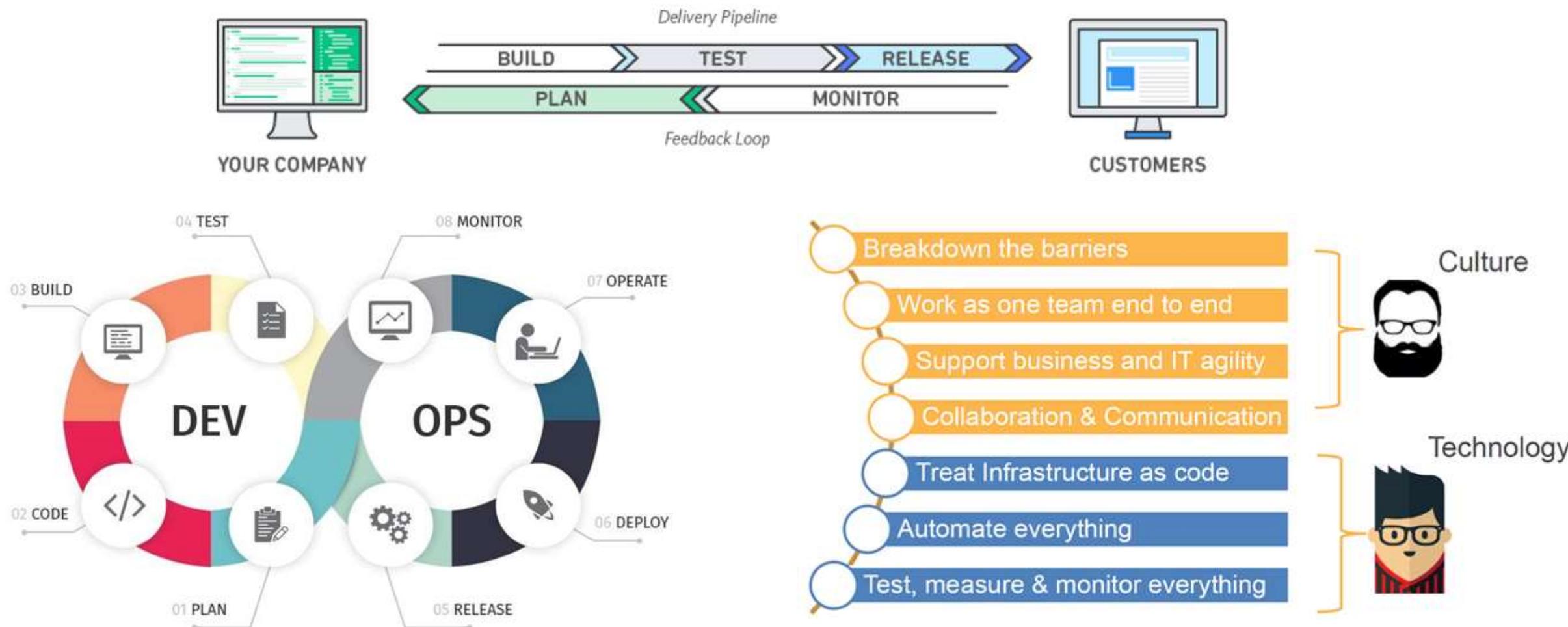
중앙 저장소(Repository)에서 코드와 이미지를 관리하고 플랫폼에 배포(Deployment)하는 형태로 지속적인 통합/배포(Continuous Integration/Deployment) 가능



Modified image from <https://medium.com/swlh/kubernetes-ci-cd-using-jenkins-on-google-cloud-5b10da6147a6>

# 하이브리드/멀티클라우드 운영 – DevOps

DevOps는 애플리케이션과 서비스를 빠른 속도로 제공할 수 있도록 조직의 역량을 향상시키는 문화 철학, 방식 및 도구의 조합



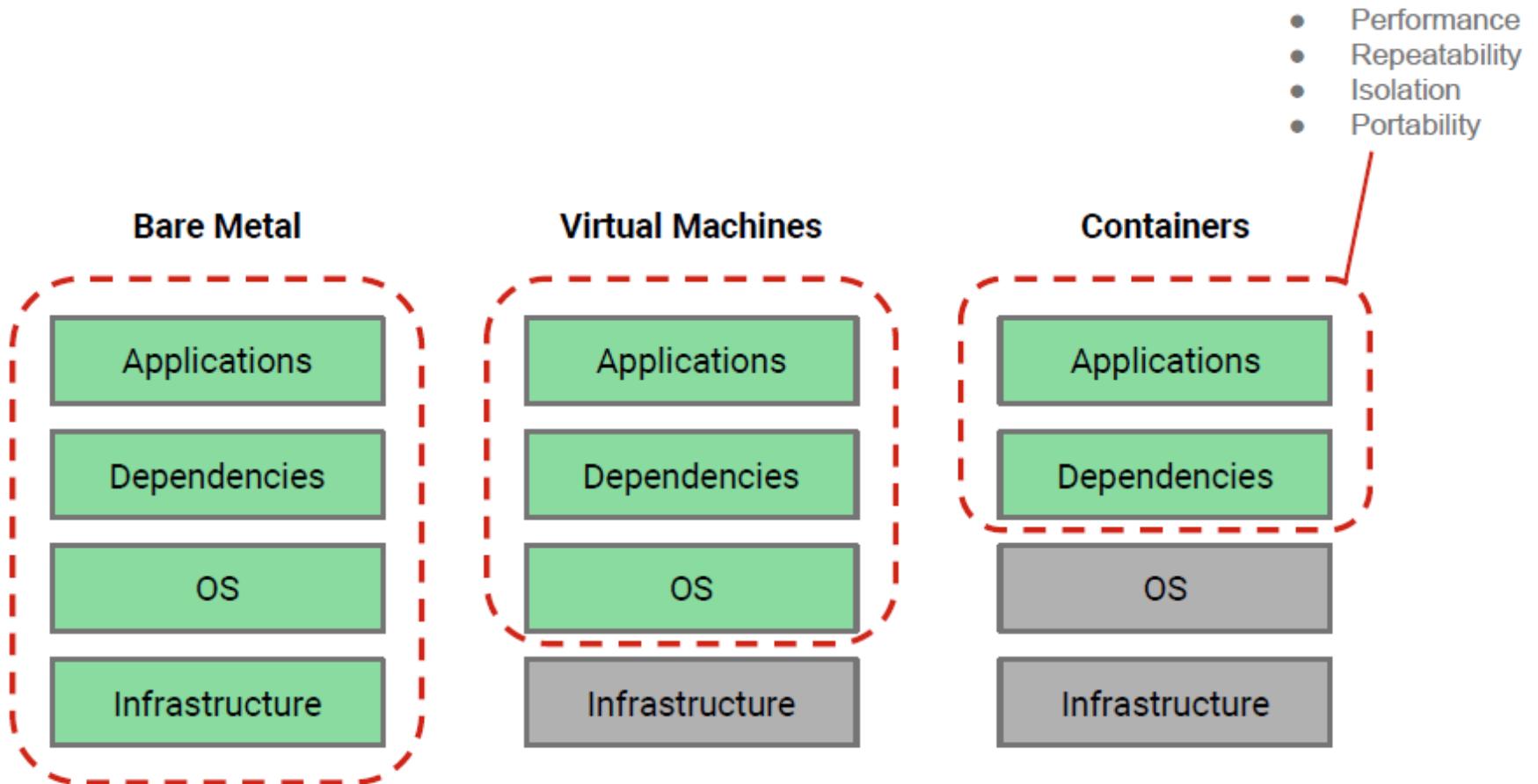
<https://aws.amazon.com/ko/devops/what-is-devops/>  
<https://dev.to/ashokisaac/devops-in-3-sentences-17c4>

# **Part II**

# **Docker Container**

# 컨테이너란 ?

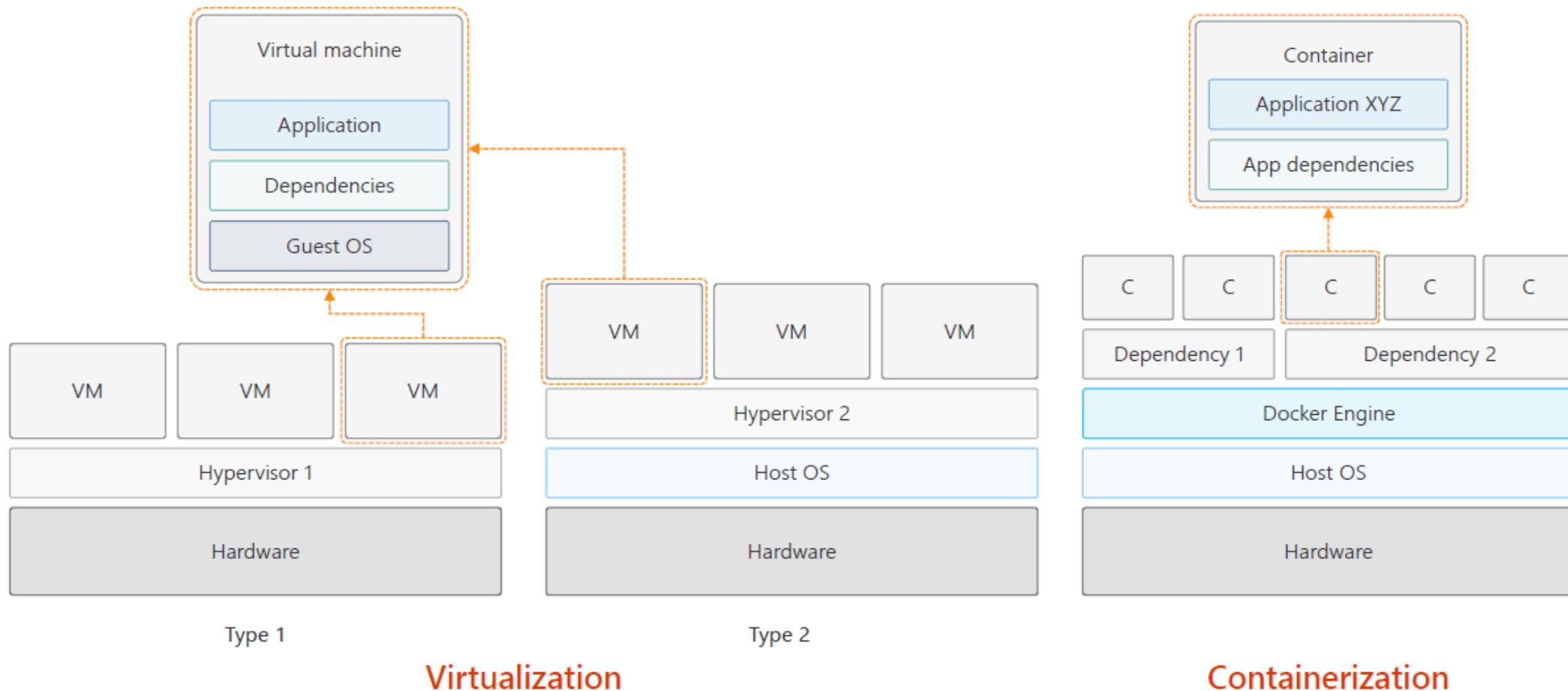
Containers are a **method of packaging** an application executable and its dependencies (runtime, system tools, system libraries, configuration), and **running the package as a set of resource-isolated processes**



# 컨테이너 기반 솔루션의 장점

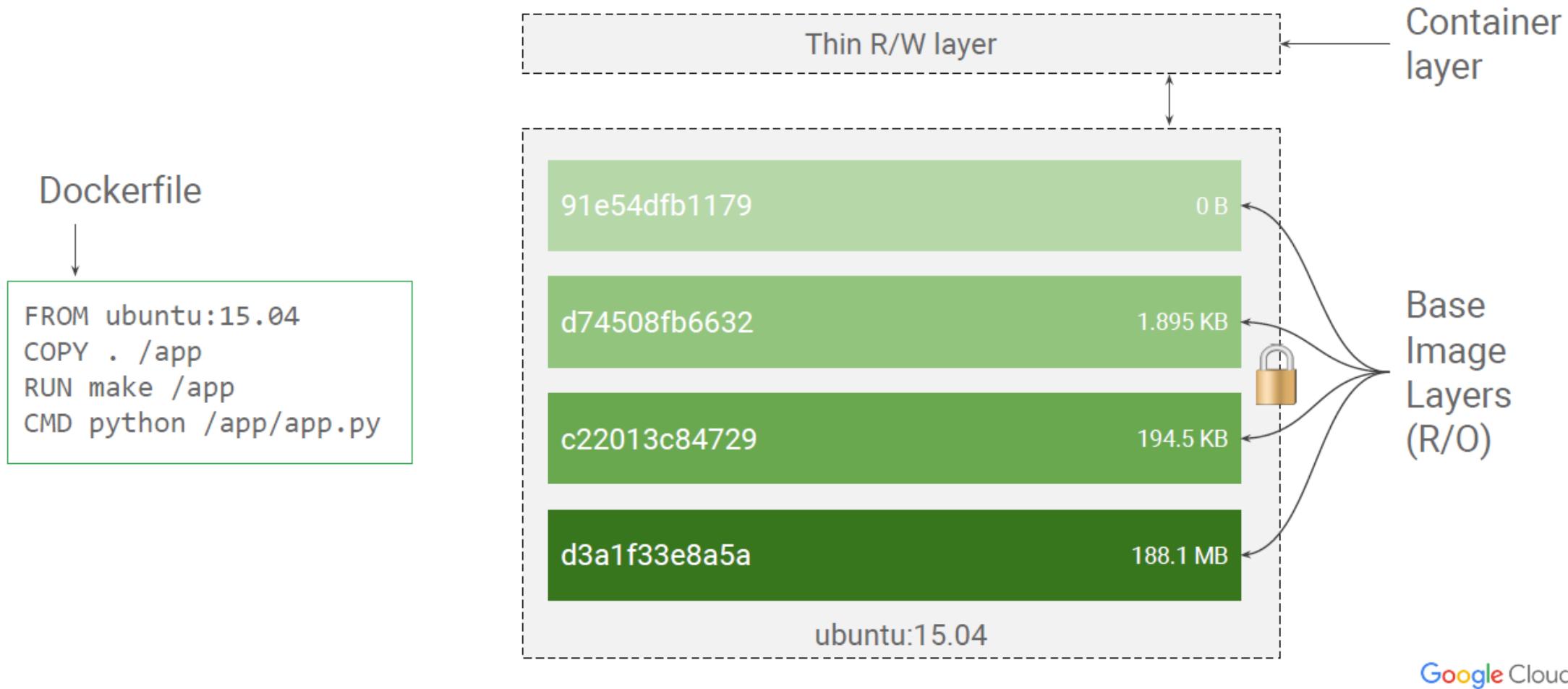
- Manage applications, not machines
- Maintain vendor independence
- Write once, run anywhere
  - Develop application on premise
  - Upload to cloud for production and scale
- Workload relocatability
  - Migrate to a new platform
- Decouple applications from dependencies
  - Ex: Namespaces, services, DNS, Secrets, specific APIs

# 가상화 vs. 컨테이너화



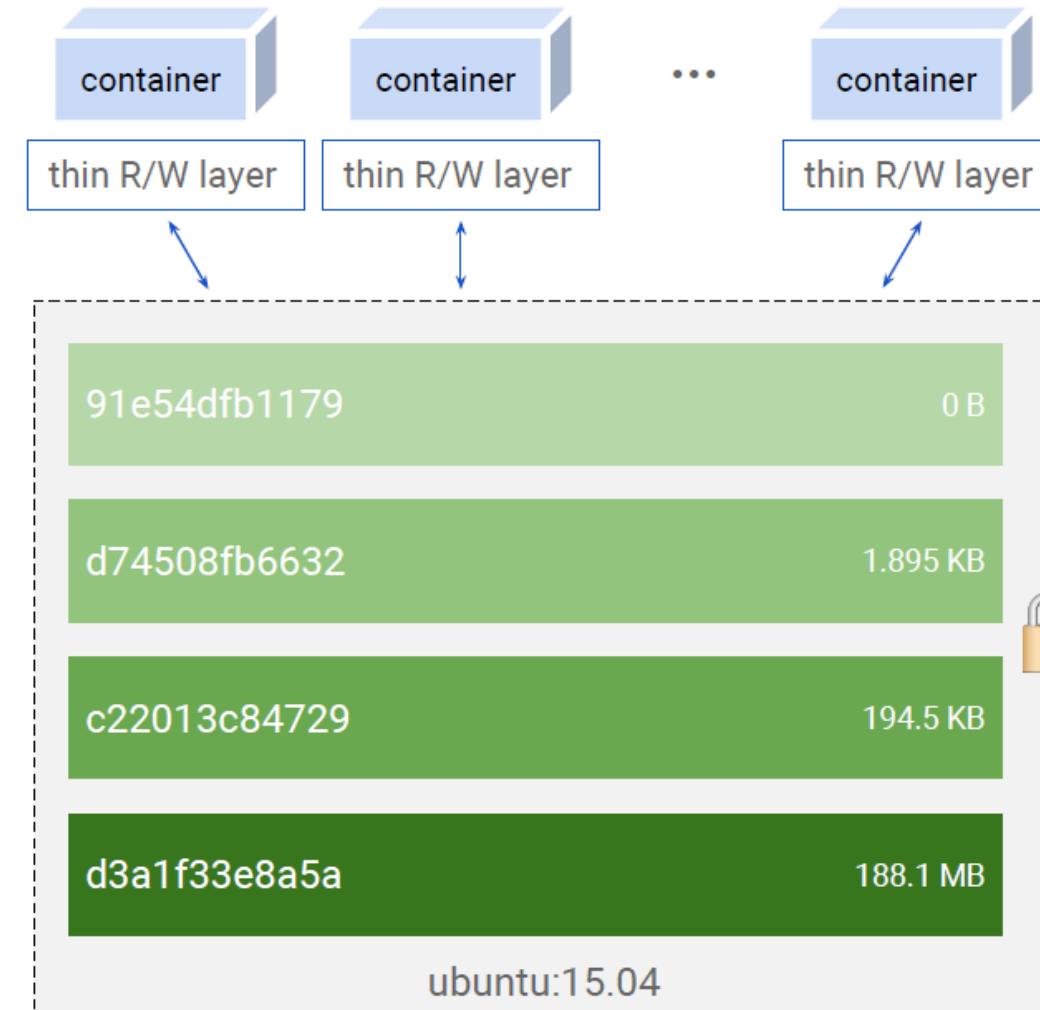
# 컨테이너는 Top Layer에만 Writing이 가능

Containers use a layered file system with only the top layer writable

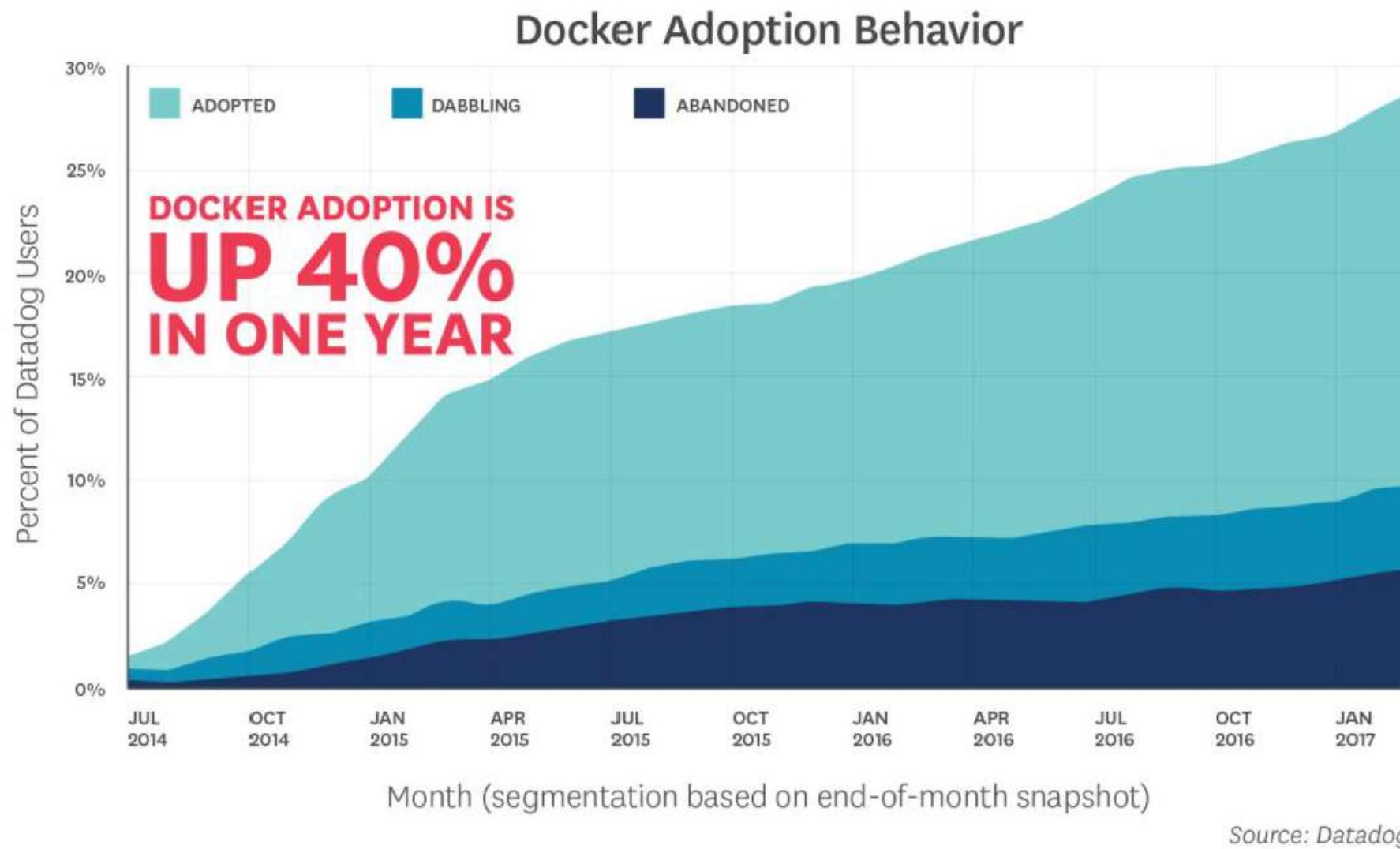


# 컨테이너는 공유된 이미지의 더 작은 부분만을 사용할 수 있게 함

Containers promote smaller shared images

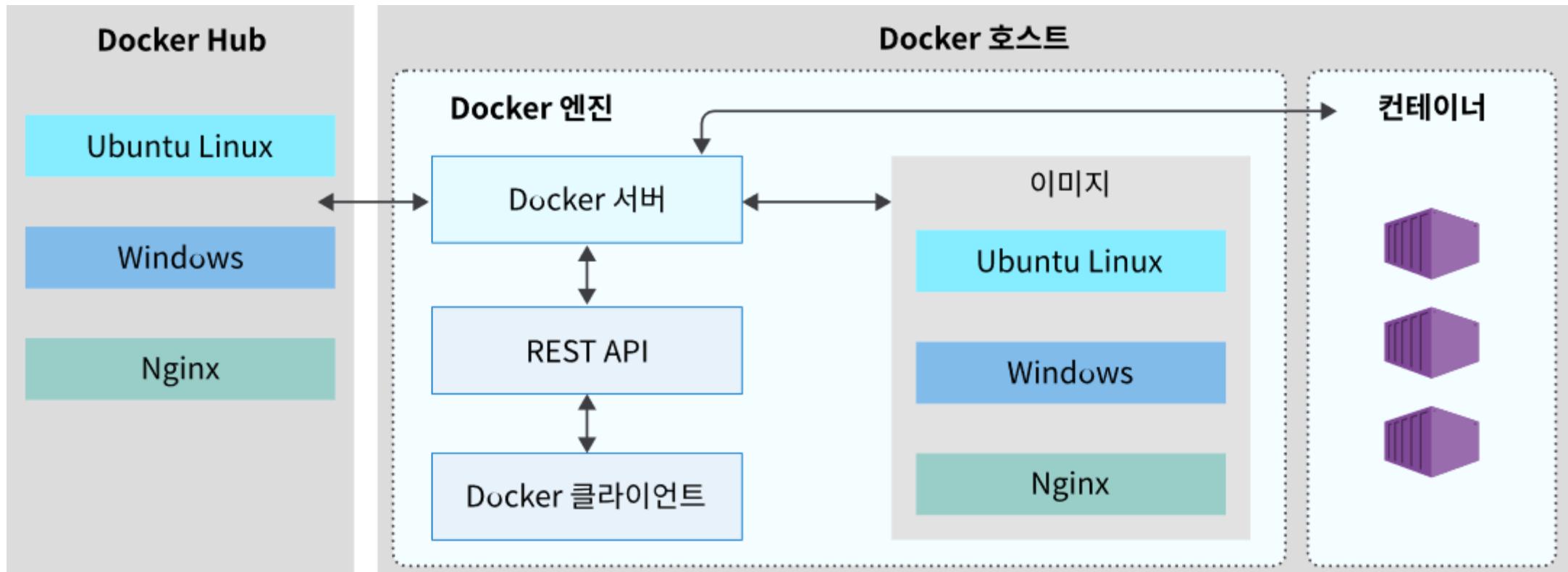


# 도커 컨테이너의 확산



# 도커 컨테이너 소개

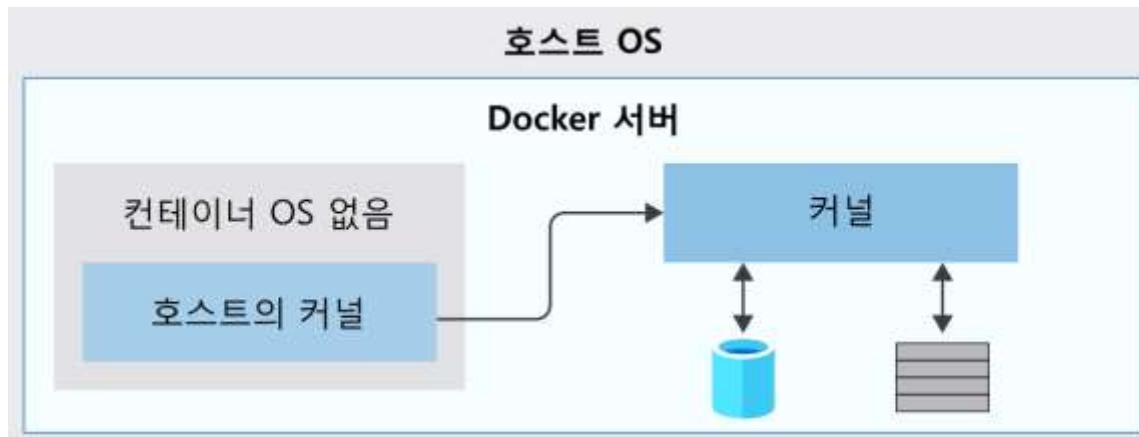
- 도커 아키텍처 (Docker Architecture)



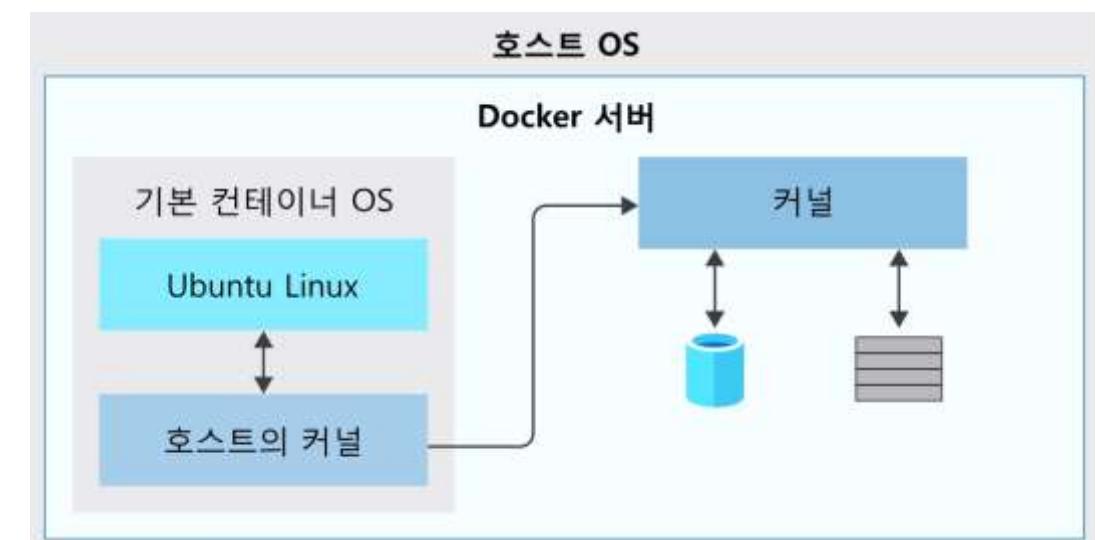
# 도커 컨테이너 소개

- 컨테이너 이미지 : 소프트웨어를 포함하는 이식 가능 패키지. 메모리 내에서 실행할 경우 컨테이너가 됨. 이미지는 변경 불가능.
- 호스트 OS: Docker 엔진이 실행되는 OS. Linux에서 실행되는 Docker 컨테이너는 호스트 OS 커널을 공유하며 이진 파일이 OS 커널에 직접 액세스할 수 있는 한 컨테이너 OS가 필요하지 않음.
- 컨테이너 OS: 패키지된 이미지에 포함된 OS. 컨테이너에는 여러 버전의 Linux 또는 Windows OS를 유연하게 포함할 수 있음. 컨테이너 OS는 호스트 OS와 격리되며 애플리케이션을 배포하고 실행하는 환경.

Host OS 설명



Container OS 설명



# 도커 컨테이너 소개

- 스택 가능 통합 파일 시스템(Unionfs) : Docker 이미지를 만드는데 사용되어, 여러 콘텐츠들이 위에 쌓일 수 있도록 하는 파일 시스템. 이미지는 제거할 수는 없고 추가할 수만 있음.
- 기본 이미지 (Base Image): Docker scratch 이미지를 사용하는 이미지. 파일 시스템 레이어를 만들지 않는 빈 컨테이너 이미지. 호스트 OS 커널을 직접 사용할 수 있다고 가정. Dockerfile 내에 FROM scratch 명령을 넣어서 만들 수 있음.
- 부모 이미지 (Parent Image): 이미지를 만드는 원본 컨테이너 이미지. 일반적으로 부모 이미지에는 컨테이너 OS가 포함.



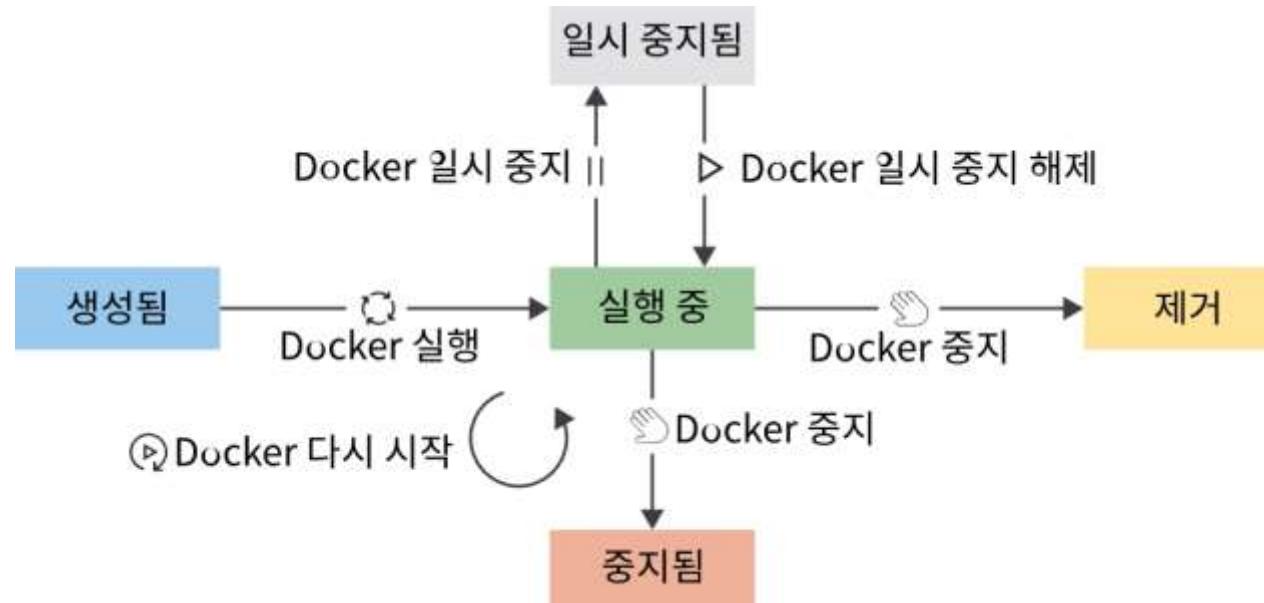
# 도커 컨테이너 소개

- Dockerfile : Docker 이미지를 빌드하고 실행하는 데 사용하는 지침을 포함하는 텍스트 파일.
  - 새 이미지를 만드는 데 사용하는 기본 또는 부모 이미지
  - 기본 OS를 업데이트하고 추가 소프트웨어를 설치하는 명령
  - 개발된 애플리케이션과 같이 포함할 빌드 아티팩트
  - 스토리지 및 네트워크 구성과 같이 공개할 서비스
  - 컨테이너가 시작될 때 실행할 명령

# 도커 컨테이너 작동방식

- Docker 컨테이너 관리 방법

- start 명령을 사용하여 컨테이너를 실행 상태로 전환, 혹은 이미 실행 중인 컨테이너를 재시작.
- pause 명령을 사용하여 실행 중인 컨테이너의 모든 프로세스를 일시 중지
- stop 명령을 사용하여 실행 중인 컨테이너를 중지. 프로세스가 종료되면 컨테이너의 커널이 종료.
- kill 명령을 사용하여 컨테이너를 강제로 종료.
- remove 명령을 사용하여 중지 상태의 컨테이너를 제거. 컨테이너에 저장된 모든 데이터가 제거됨.



# 도커 컨테이너 작동방식

- Docker 컨테이너 명령어
  - docker ps : 실행 중인 컨테이너를 나열. 상태에 관계없이 모든 컨테이너를 보려면 -a 플래그.

docker ps -a						
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
D93d40cc1ce9	tmp-ubuntu:latest	"dotnet website.dll ..."	6 seconds ago	Up 5 seconds	8080/tcp	happy_wilbur
33a6cf71f7c1	tmp-ubuntu:latest	"dotnet website.dll ..."	2 hours ago	Exited (0) 9 seconds ago		adoring_borg

\* 동일한 이미지에서 여러 컨테이너 인스턴스를 만드는 방법 : run --name 플래그를 사용

- docker run : 컨테이너를 실행. Background에서 실행을 하려면 -d 플래그.

```
docker run -d tmp-ubuntu
```

- docker pause & docker restart : 컨테이너를 일시 중단 & 재실행.

```
docker pause happy_wilbur
docker restart happy_wilbur
```

- docker stop & docker rm : 컨테이너를 중단 & 제거.

```
docker stop happy_wilbur
docker rm happy_wilbur
```

# 도커 컨테이너 작동방식

- Docker 컨테이너 스토리지 구성
  - 컨테이너의 애플리케이션이 데이터를 저장해야 하는 경우 항상 컨테이너는 임시 데이터를 유지하는 두 가지 옵션 - 첫 번째 옵션은 \_볼륨\_을 사용하는 것이고, 두 번째 옵션은 \_바인드 탑재\_
- 1) docker volume create 명령을 사용하여 새 볼륨을 만들고 관리. 이 명령은 Dockerfile 정의의 일부를 구성. 볼륨은 호스트 파일 시스템의 디렉터리 내 특정 폴더 위치를 지정하여 저장. 볼륨을 처음으로 컨테이너에 탑재하려고 할 때 Docker는 볼륨을 생성 (없는 경우). 탑재된 후 이 볼륨은 호스트 머신에서 격리. 여러 컨테이너가 동시에 동일한 볼륨을 사용할 수 있음. 컨테이너가 볼륨 사용을 중지하는 경우 볼륨이 자동으로 제거되지 않음.
- 2) 바인드 탑재는 개념적으로 볼륨과 동일하지만, 특정 폴더를 사용하는 대신 파일이나 폴더를 호스트에 탑재할 수 있음. 또한 호스트가 이 탑재의 콘텐츠를 변경할 수 있어야 함. 볼륨처럼 파일이나 폴더를 탑재하는데 아직 호스트에 없으면 바인드 탑재가 생성.

# 도커 컨테이너 작동방식

- Docker 컨테이너 네트워크 구성

Docker 에서는 미리 설정된 3가지 네트워크 모드가 가능 : 1) 브리지, 2) 호스트, 3) 네트워크 모드 없음

- 브리지 네트워크는 추가 네트워크 구성을 지정하지 않고 시작될 때 컨테이너에 적용되는 기본 구성. 컨테이너에서 사용되는 내부 프라이빗 네트워크이며 Docker 호스트 네트워크에서 컨테이너 네트워크를 격리. 브리지 네트워크의 각 컨테이너에는 IP 주소 및 서브넷 마스크가 할당되며 호스트 이름이 기본적으로 컨테이너 이름으로 지정. 브리지 네트워크에서는 호스트 이름을 사용한 컨테이너 간 통신을 허용하지 않음.

기본적으로 Docker는 컨테이너 포트를 게시하지 않으므로, Docker 포트 --publish 플래그를 사용하여 컨테이너 포트와 Docker 호스트 포트 간에 포트 매핑을 사용하도록 설정.

예) 포트 80을 검색하는 클라이언트에서 액세스 하도록, 컨테이너의 포트 80을 호스트의 사용 가능하고 열려 있는 포트 8080에 플래그를 설정.

```
--publish 80:8080
```

- 호스트 네트워크에서 직접 컨테이너를 실행. 네트워크 수준에서 호스트와 컨테이너 간 격리를 효과적으로 제거. 계속 호스트 IP를 사용하여 액세스할 수 있으며, 매핑된 포트를 사용할 필요는 없지만, 호스트에서 사용되지 않는 포트만 사용 가능.
- 컨테이너의 네트워킹을 사용하지 않도록 설정.

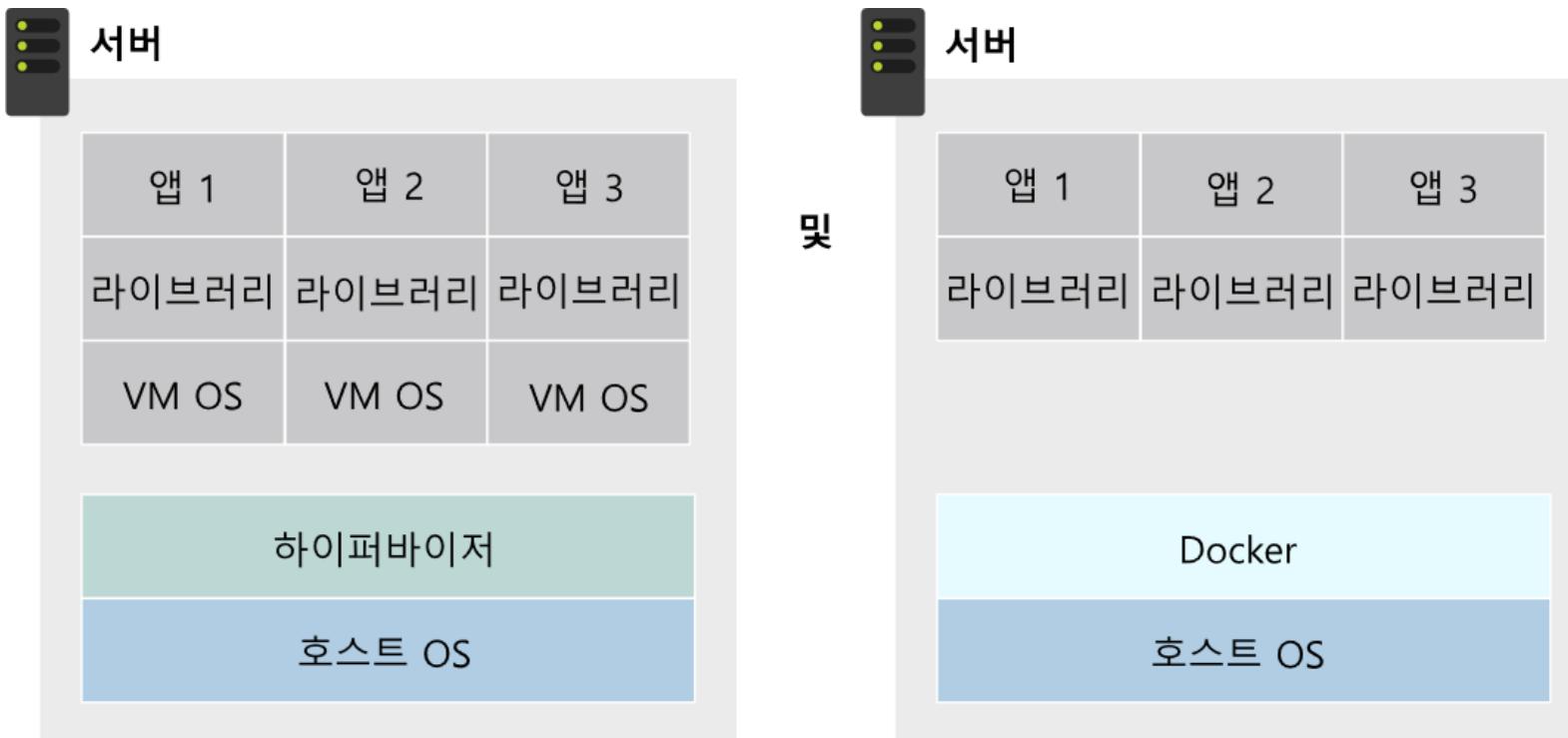
# Docker 컨테이너를 사용하는 경우

- Docker 이점
  - 하드웨어의 효율적인 사용



# 도커 컨테이너 작동방식

- Docker 이점
  - 컨테이너 격리



# 도커 컨테이너 작동방식

- Docker 이점
  - 애플리케이션 이동성  
컨테이너는 거의 모든 위치, 데스크톱, 물리적 서버, VM 및 클라우드에서 실행되는 런타임 호환성 덕분에 서로 다른 환경 간에 컨테이너화된 애플리케이션을 쉽게 이동가능. VM과 같은 느린 시작 또는 종료 시간 문제가 없음.
  - 애플리케이션 제공  
Docker를 사용하면 컨테이너는 애플리케이션을 배포하는 데 사용하는 단위로 작용하여, 개발 팀이 애플리케이션 빌드를 릴리스한 후 배포 시스템의 모든 단계에서 컨테이너를 사용할 수 있음. 컨테이너는 연속 통합에 적합한 후보이며 빌드에서 프로덕션까지 시간을 단축.
  - 호스팅 환경 관리  
애플리케이션 환경은 컨테이너에 내부적으로 구성되어, 운영 팀이 OS 업데이트를 모니터링하고, 보안 패치를 한 번 적용하고, 필요에 따라 업데이트된 컨테이너를 출시하는 유연성을 제공.
  - 클라우드 배포  
많은 클라우드 플랫폼에서 지원되어 인프라를 관리하는 오버헤드 없이 애플리케이션 디자인 및 빌드에 집중 가능하게 함. Container-based managed VM, Kubernetes, Serverless 등 많은 경우에 적용하여 배포 가능.

# Docker를 사용하여 컨테이너화된 웹 애플리케이션 빌드

- 도커 컨테이너를 이용한 App Build 개요

Docker 이미지로 배포하고 Azure Container Instance에서 실행할 수 있도록 웹앱을 패키징

- Docker 허브의 시작 이미지를 기반으로 새 컨테이너 이미지의 Dockerfile 만들기
- Dockerfile 명령을 사용하여 이미지에 파일 추가
- Dockerfile 명령을 사용하여 이미지의 시작 명령 구성
- Docker 이미지에 패키징된 웹 애플리케이션을 빌드하고 실행
- Azure Container Instance 서비스를 사용하여 Docker 이미지 배포

# Docker를 사용하여 컨테이너화된 웹 애플리케이션 빌드

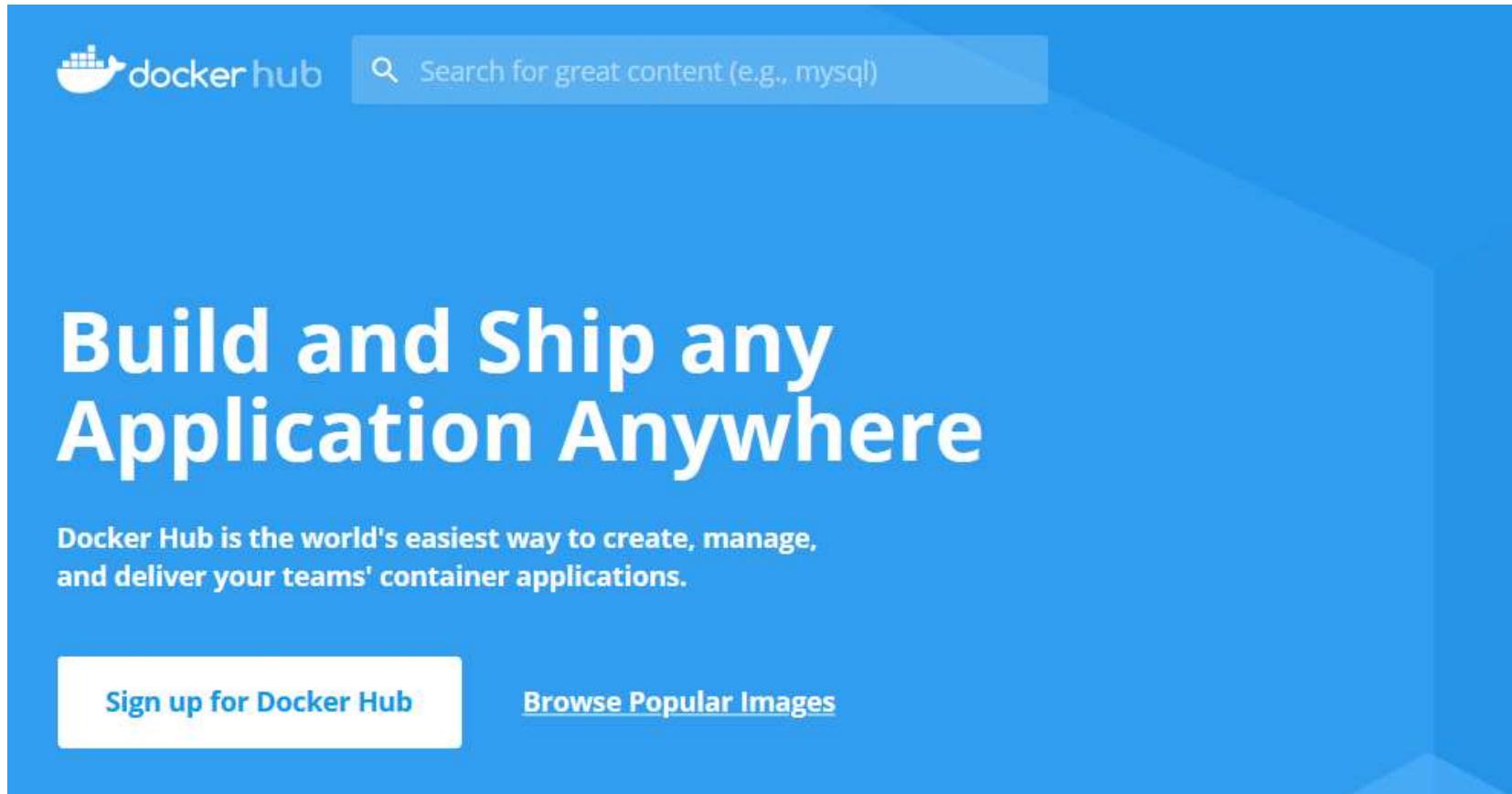
- 기존 Docker 이미지를 검색하여 로컬로 배포

Azure Container Registry에 Docker 이미지를 저장하고 Azure Container Instances를 사용하여 컨테이너를 실행

- 개별 Docker 이미지는 Windows 기반으로 또는 Linux 기반으로 사용할 수 있지만, 둘 다 동시에 사용 할 수는 없습니다. 이미지의 운영 체제에 따라 컨테이너 내에서 사용할 운영 체제 환경의 종류가 결정됨.
- Windows 기반 이미지와 Linux 기반 이미지에서 비슷한 기능을 제공하고 싶은 Docker 이미지 작성자는 각각의 이미지를 별도로 빌드.
- Docker가 설치된 Linux 컴퓨터는 Linux 컨테이너만 실행할 수 있으나, Docker가 설치된 Windows 컴퓨터는 두 가지 컨테이너를 모두 실행할 수 있음. Windows가 둘 다 실행할 수 있는 이유는 가상 머신을 사용하여 Linux 시스템을 실행하고, 가상 Linux 시스템을 사용하여 Linux 컨테이너를 실행하기 때문.

# 기존 Docker 이미지 검색

<https://hub.docker.com/>



# **Part III**

# **Kubernetes**

# 쿠버네티스란 ?

- An open source project
- Framework for container management and automation
- Based on Google's systems
- Developing rapidly - complex
  - *Only covering the basics in this class*
  - *Only covering Kubernetes Engine in this class*
- More information
  - [kubernetes.io](https://kubernetes.io) (also, k8s.io)



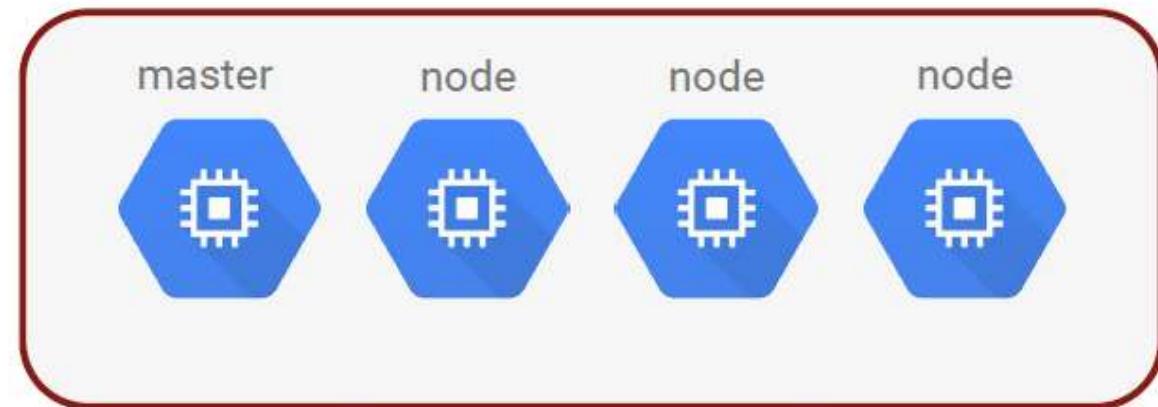
# 쿠버네티스 엔진

- Fully-managed service
  - Kubernetes software maintained
  - SLA
- Docker format containers
- Autoscaling (CPU or memory)
- Stackdriver logging and monitoring
- Cloud VPN integration
  - Hybrid cloud and on premise solutions
- Cloud IAM integration

# 컨테이너 클러스터

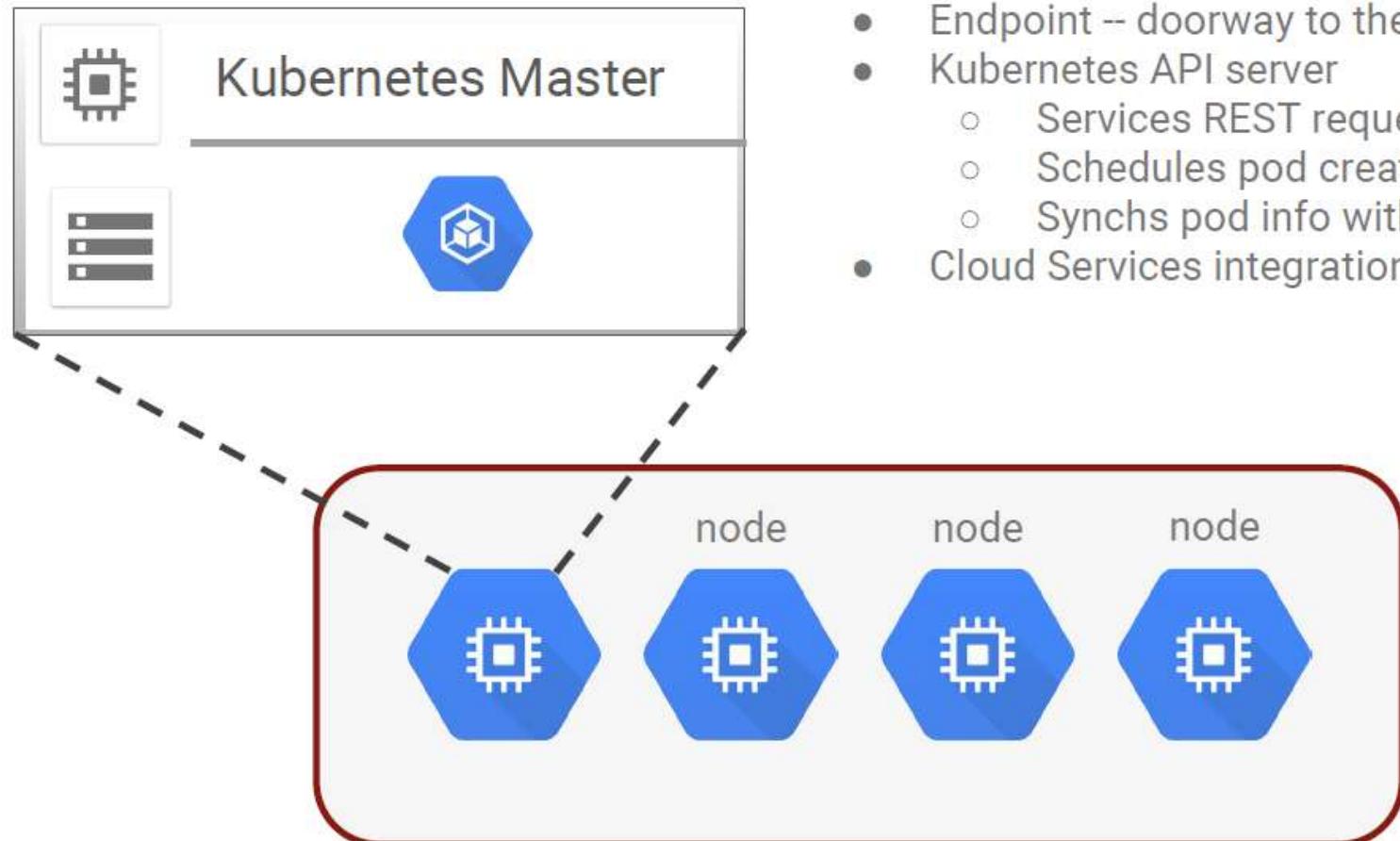
Each node runs:

- Docker runtime
- Kubelet agent
  - Manages scheduled Docker containers
- Network proxy



Container Cluster - a group of Compute Engine instances running Kubernetes

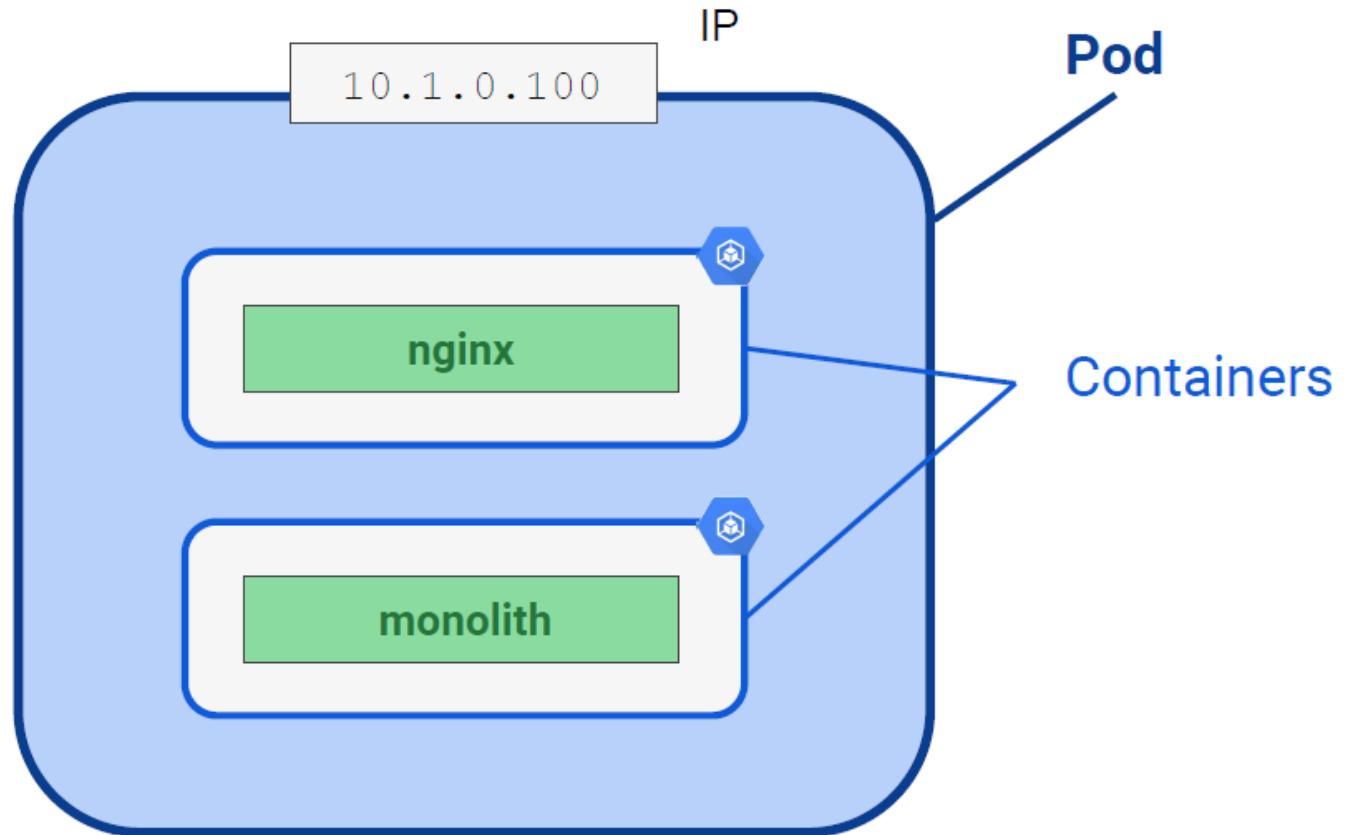
# Kubernetes Master Endpoint



- Endpoint -- doorway to the cluster
- Kubernetes API server
  - Services REST requests
  - Schedules pod creation/deletion on nodes
  - Synchs pod info with service info
- Cloud Services integration

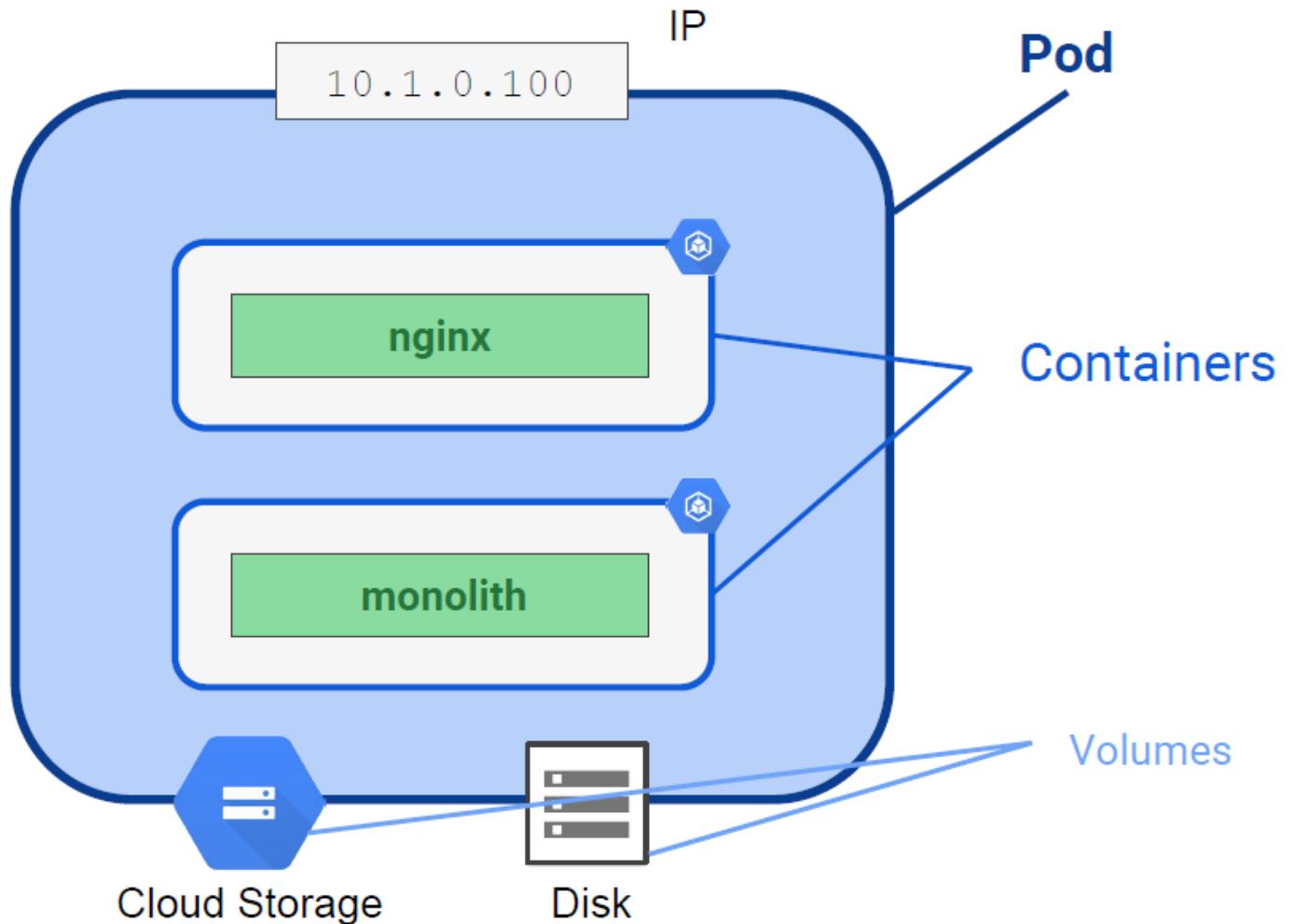
# Kubernetes Master Endpoint

- A Pod is Kubernetes Engine's abstraction to represent an application
- It holds one or more containers
- The containers in the pod share:
  - A single IP address
  - A single namespace

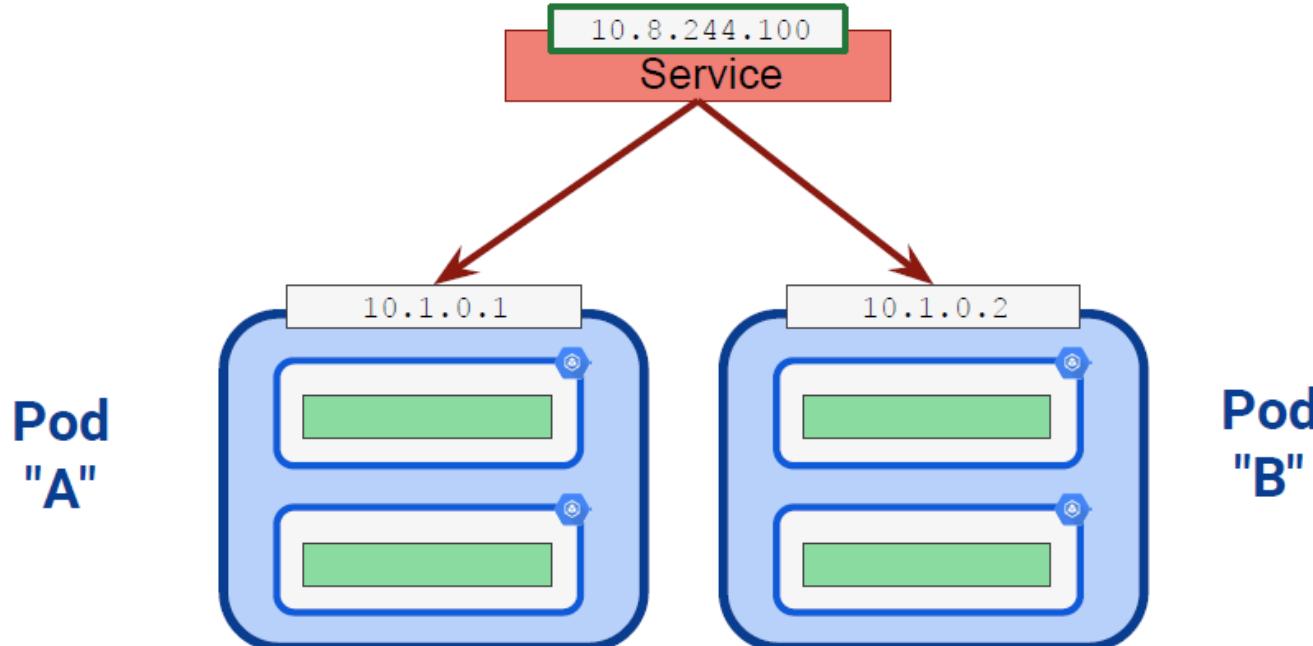


# Pods

- A Pod can share other items
  - Access to storage

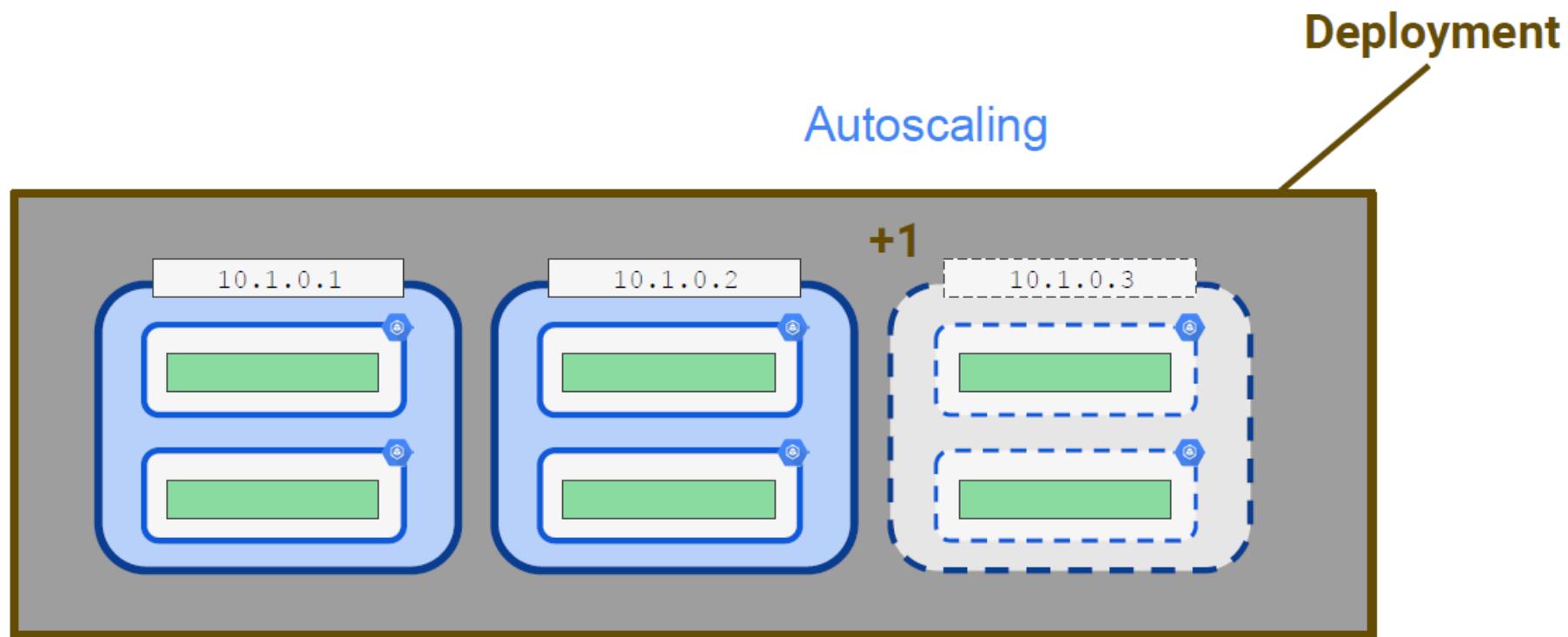


# Kubernetes Master Endpoint

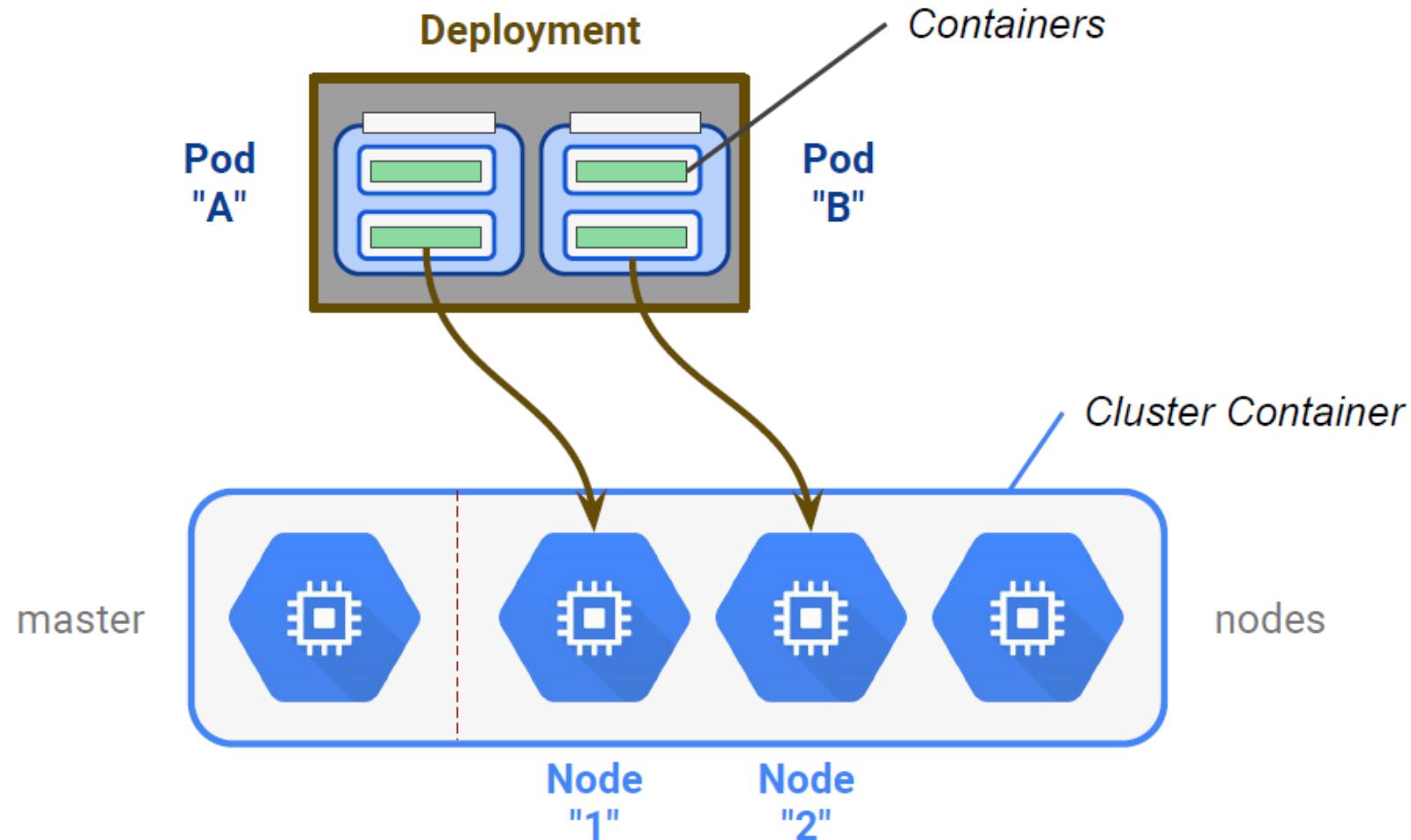


A service provides a persistent internal or external IP for pods

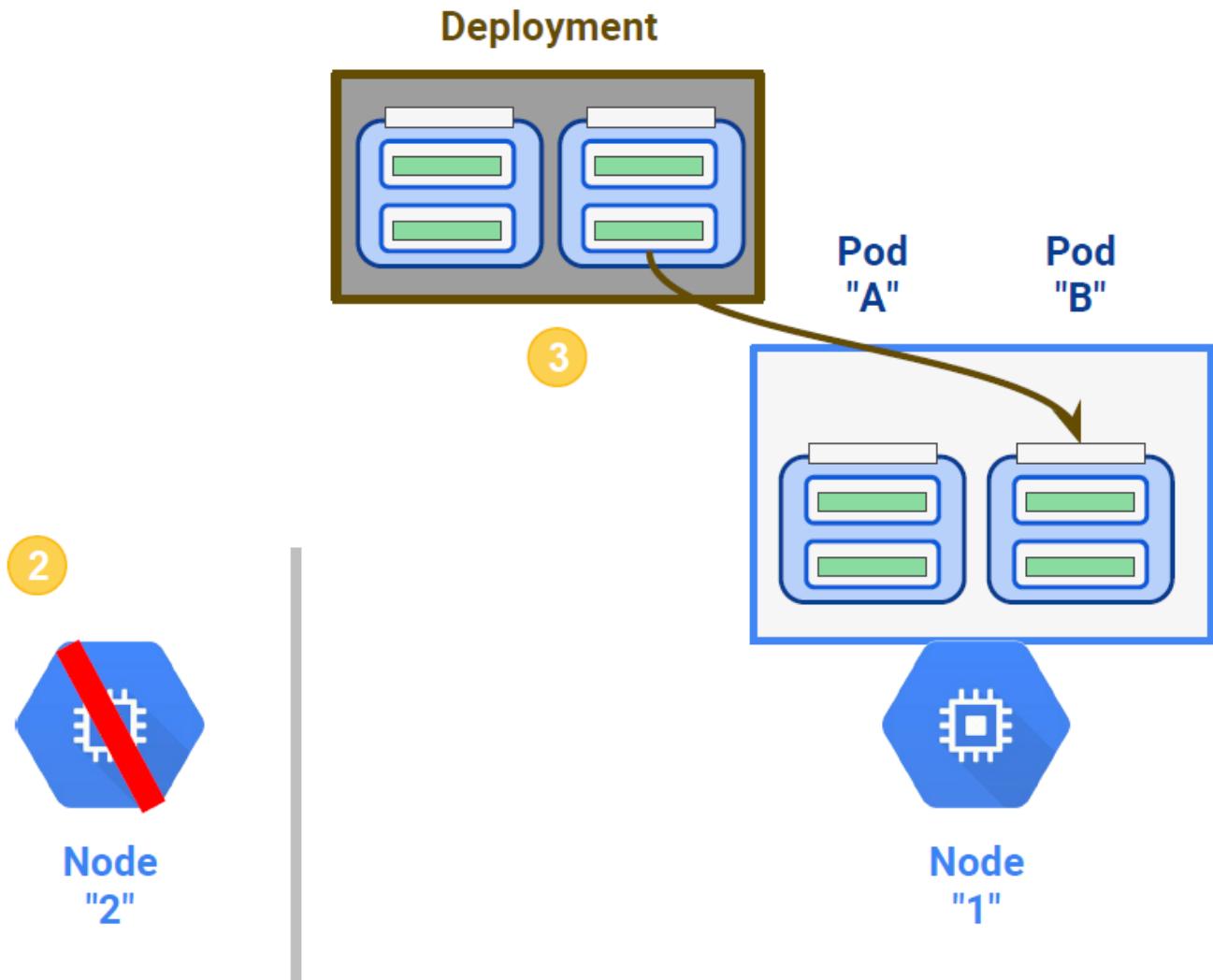
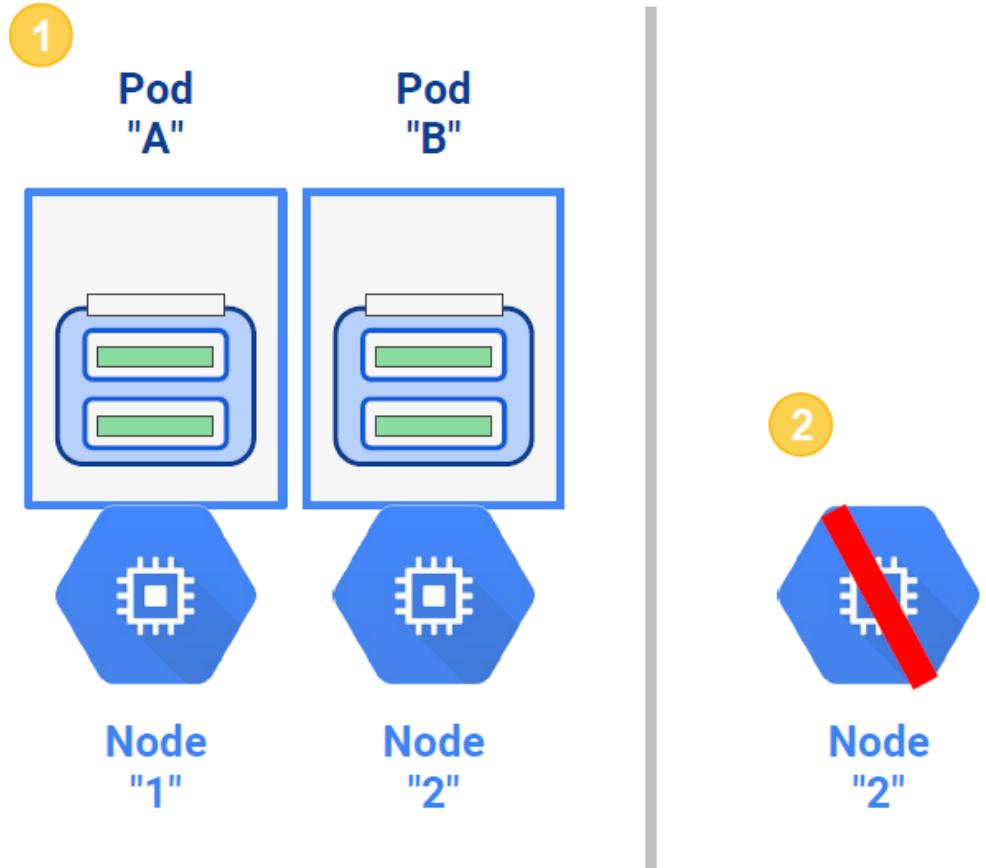
# Kubernetes Engine Deployment



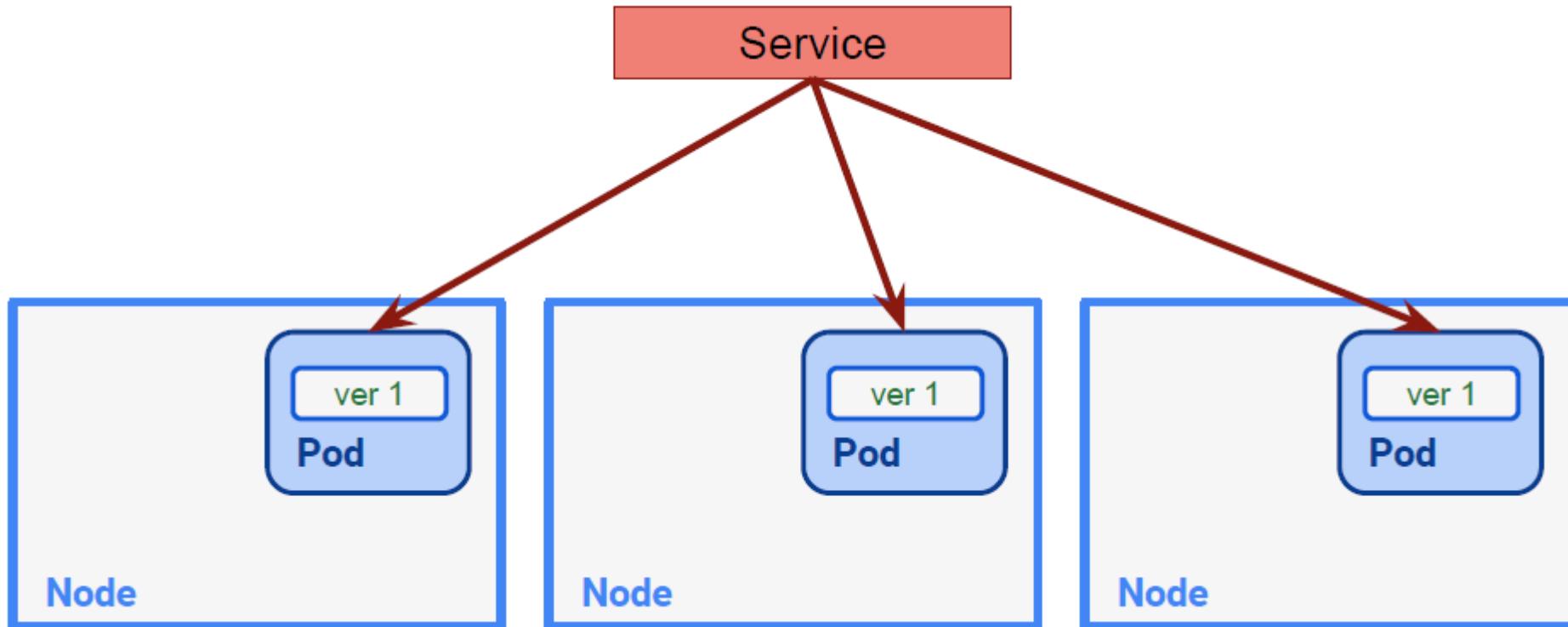
# Kubernetes Master Endpoint



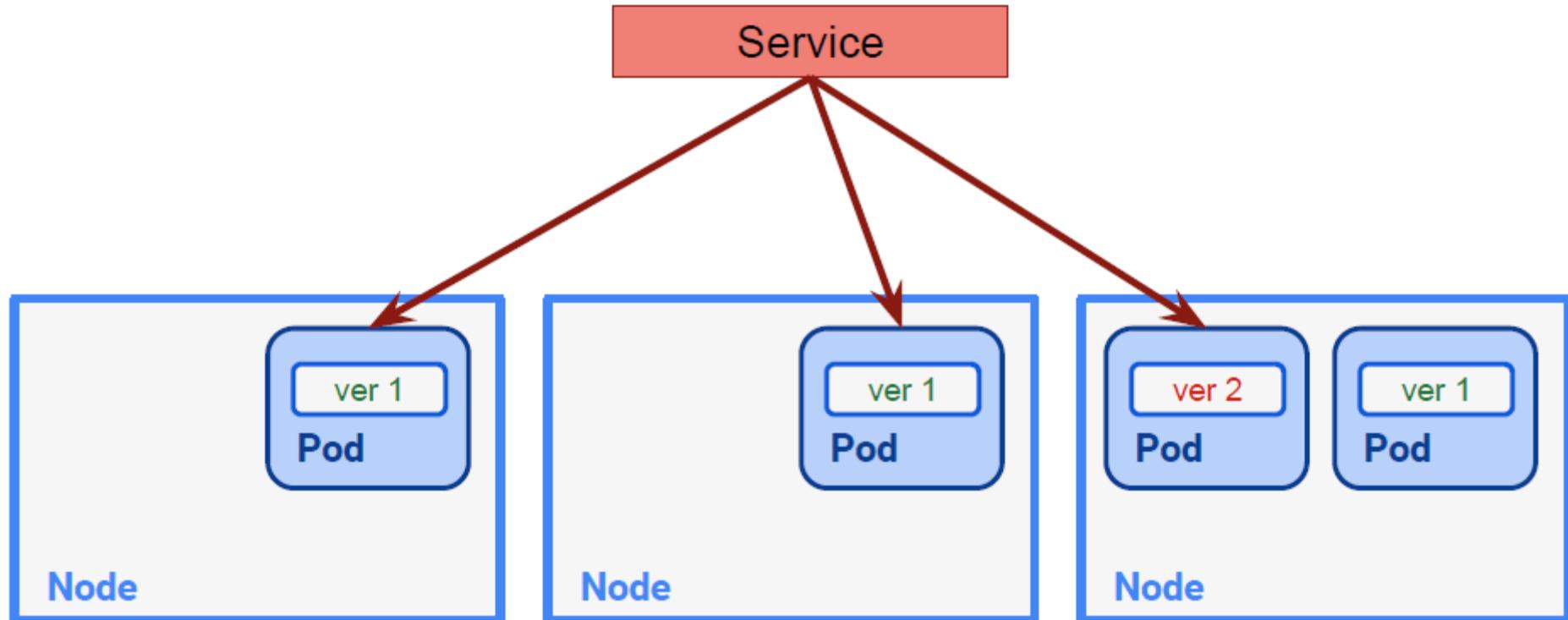
# Resiliency (복원 탄성력)



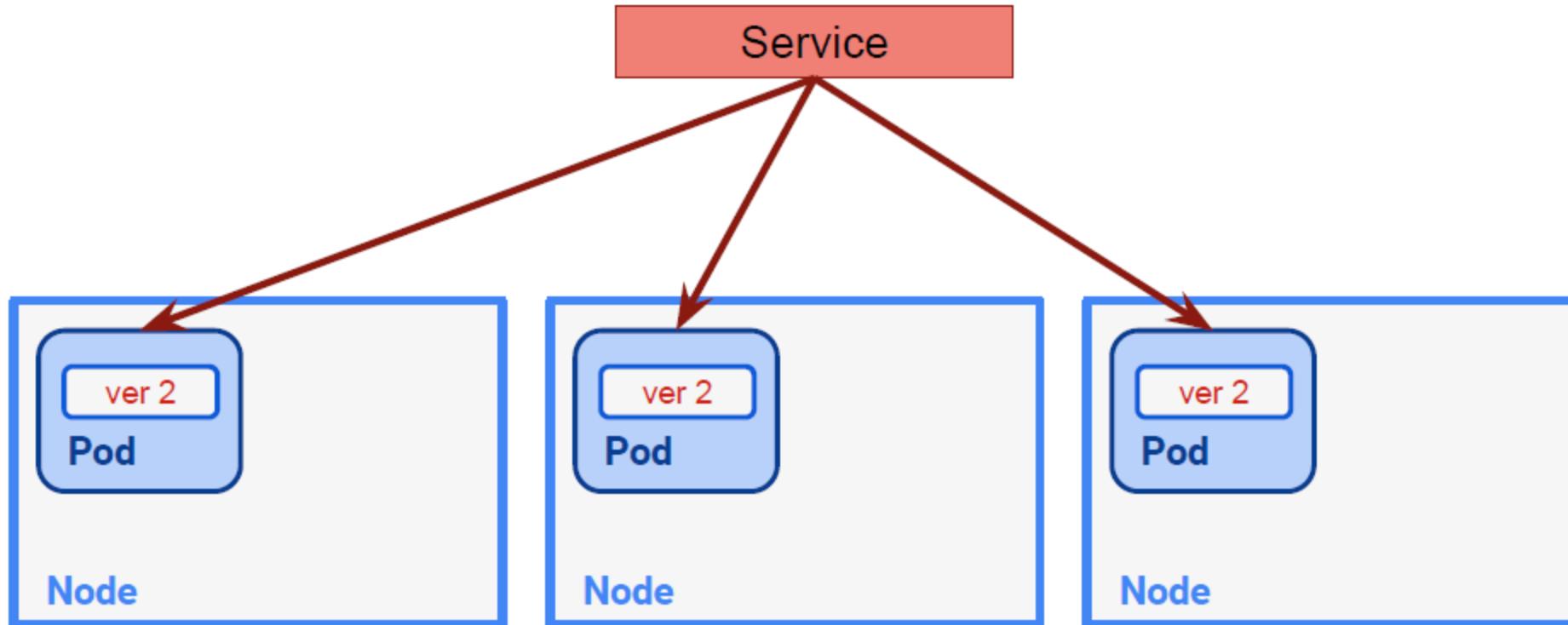
# Rolling Update (1/3)



## Rolling Update (2/3)



## Rolling Update (3/3)



## Module 3

# GCP GKE를 이용한 Kubernetes 배포운영 이해와 실습

# **Part I**

# **Google Cloud Platform (GCP) 소개**

# 구글클라우드 특장점

The screenshot shows the Google Cloud homepage in Korean. At the top, there's a navigation bar with links like 'Google Cloud', 'Google을 선택해야 하는 이유' (Why Google Cloud), '솔루션' (Solutions), '제품' (Products), '가격 책정' (Pricing), '시작하기' (Get Started), and '영업팀에 문의' (Contact Sales). Below the navigation is a search bar and user account options ('Language', '로그인', '무료로 시작하기'). The main content area features a large '획기적인 솔루션, 혁신적인 노하우' (Innovative Solutions, Innovative Techniques) heading with a small house icon. A sub-section below it discusses Google Cloud's comprehensive solutions and technologies. To the right are two 3D cube diagrams: one with a red sphere and another with a yellow sphere.

- 세계 최고의 보안 수준으로 위험 감소
- 하이브리드 및 다중 클라우드를 통한 선택권 확대
- 간편성을 위한 완전관리형 서비스 전환
- AI 및 ML을 통한 혁신
- 완전한 클라우드 서비스 제품군을 통한 현대화
- 관리형 오픈소스 소프트웨어를 사용한 개발
- Google의 세계적인 네트워크 활용

<https://cloud.google.com/why-google-cloud/>

# 구글클라우드 인프라스트럭처

제공 지역:

20

리전

61

영역

134

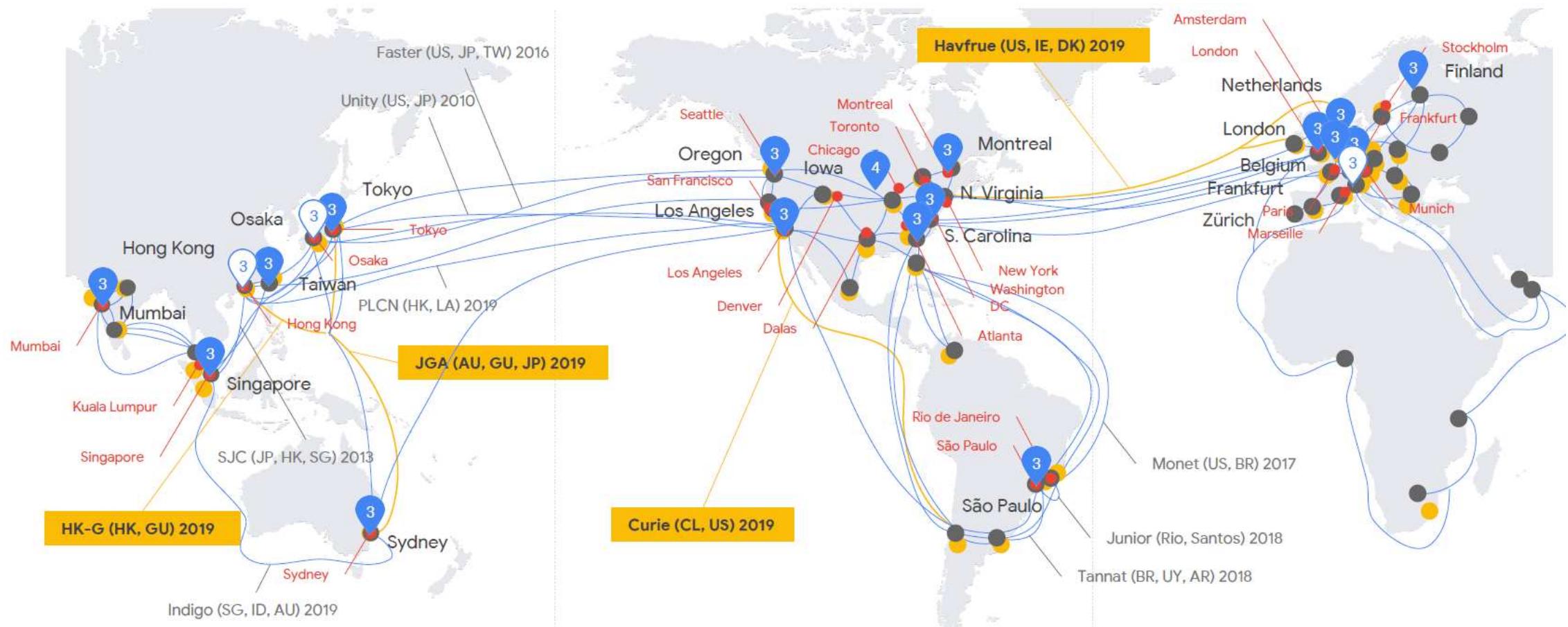
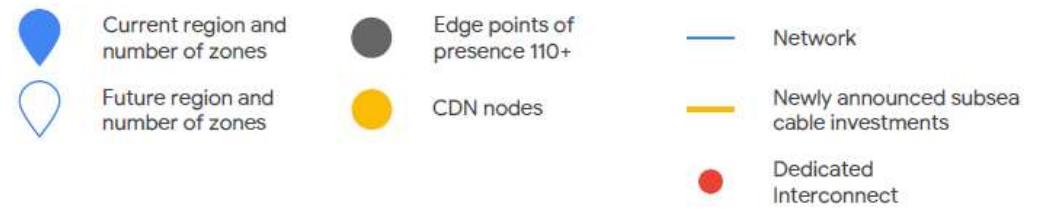
네트워크 에지 위치

200+

국가 및 지역



# 구글클라우드 인프라스트럭처



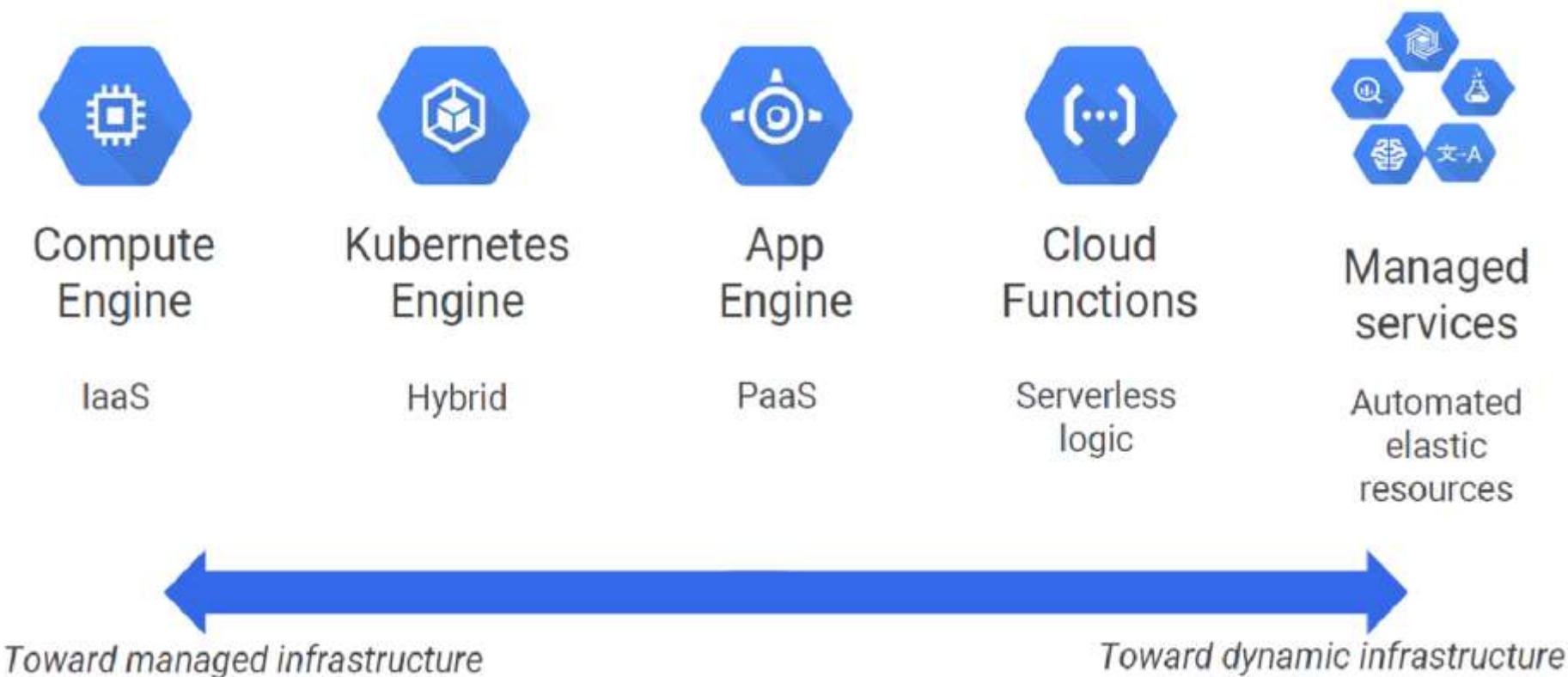
Asia Pacific

Americas

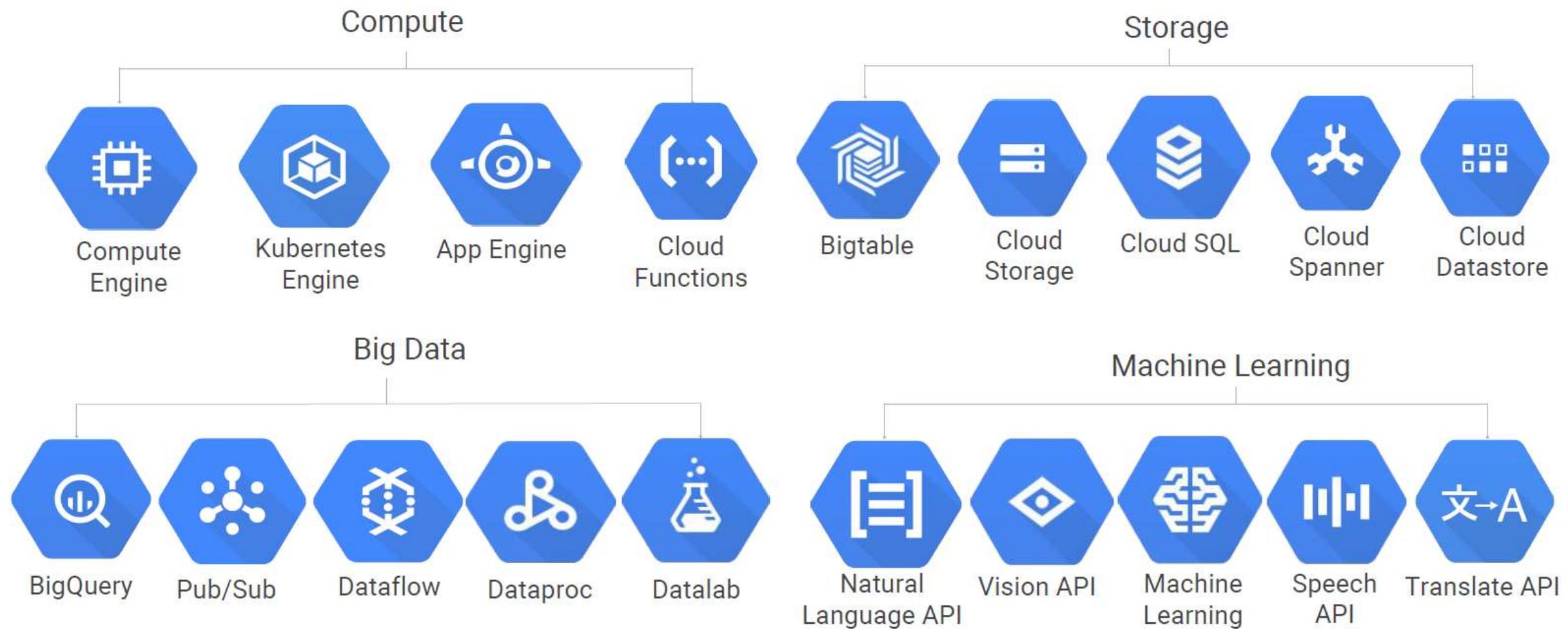
Europe, Middle East, & Africa

# 구글클라우드 제공 서비스 분류

- Managed Service 제공 정도에 따른 분류
  - IaaS (Infrastructure as a Service)
  - PaaS (Platform as a Service)
  - SaaS (Software as a Service)



# 구글클라우드 핵심 서비스 종류

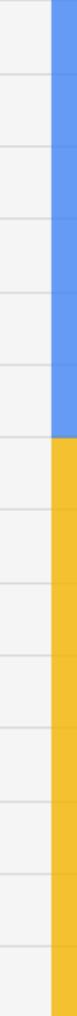
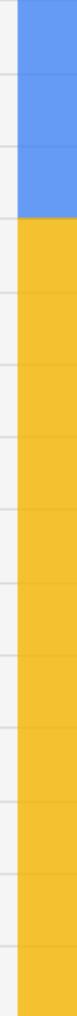
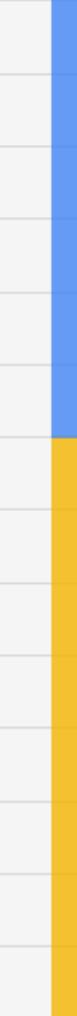
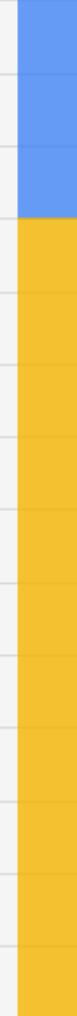
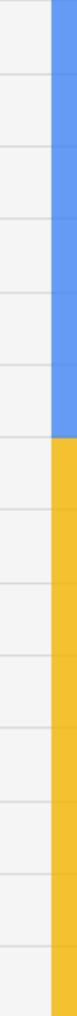
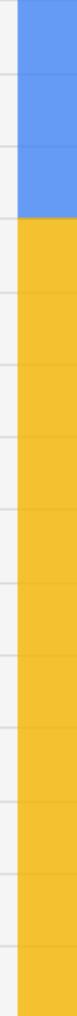
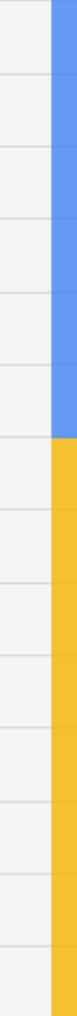
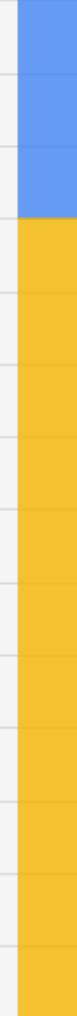
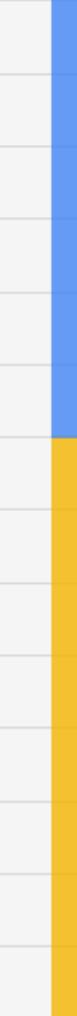
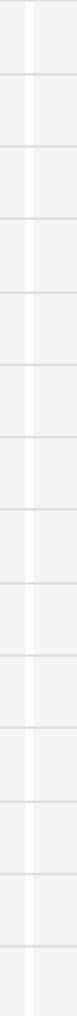
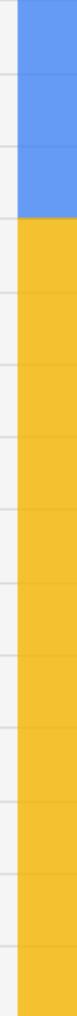
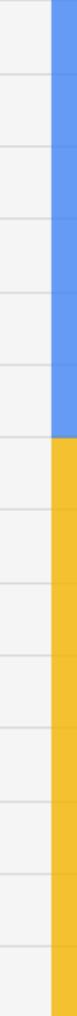
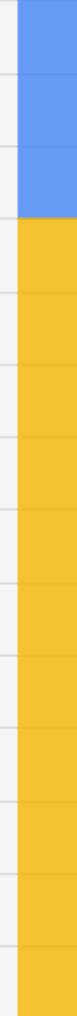
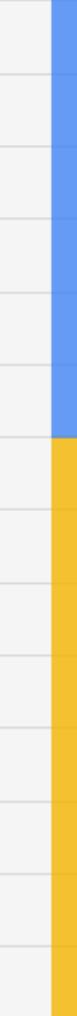
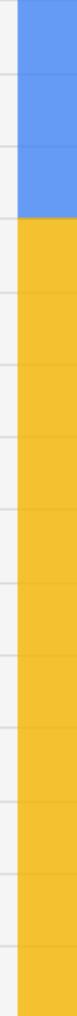
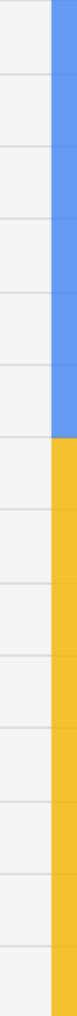
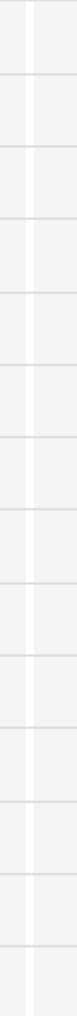
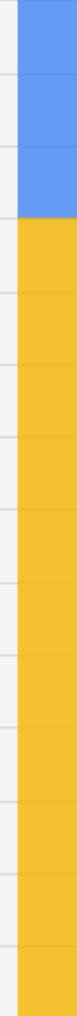
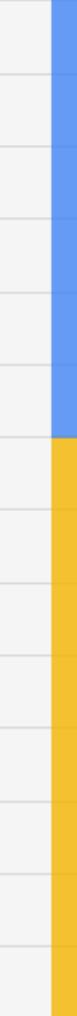
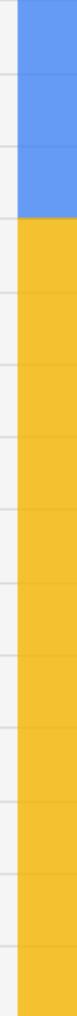
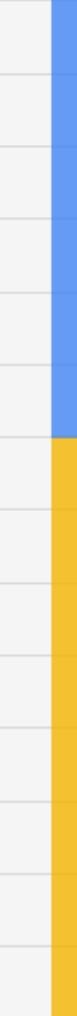
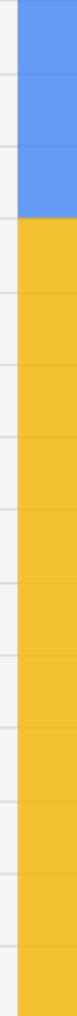
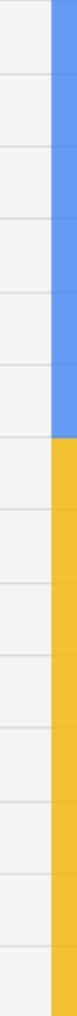
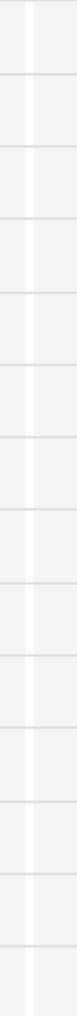
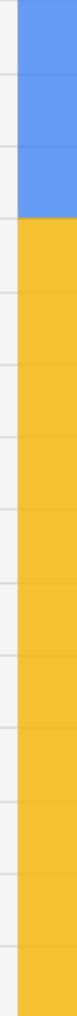
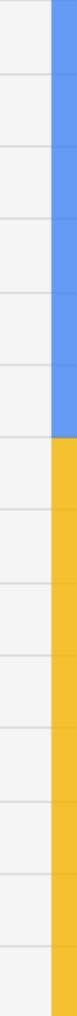
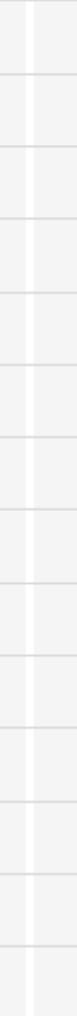
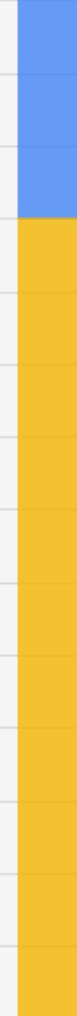
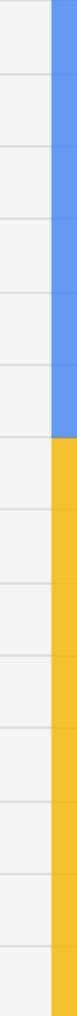
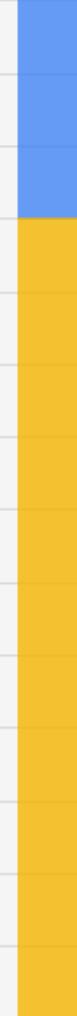
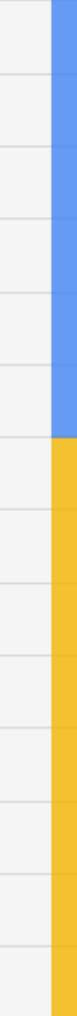
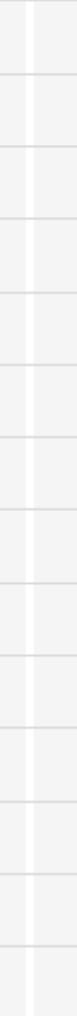
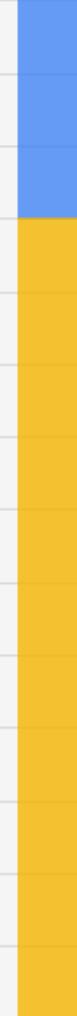


# 구글클라우드 보안 협력

## Security Collaboration

- Google은 infrastructure 보안에 책임
- 사용자는 데이터 보안에 책임
- Google은 사용자에게 best practices, templates, 제품, 솔루션 등을 제공하여 보안을 도와줌.

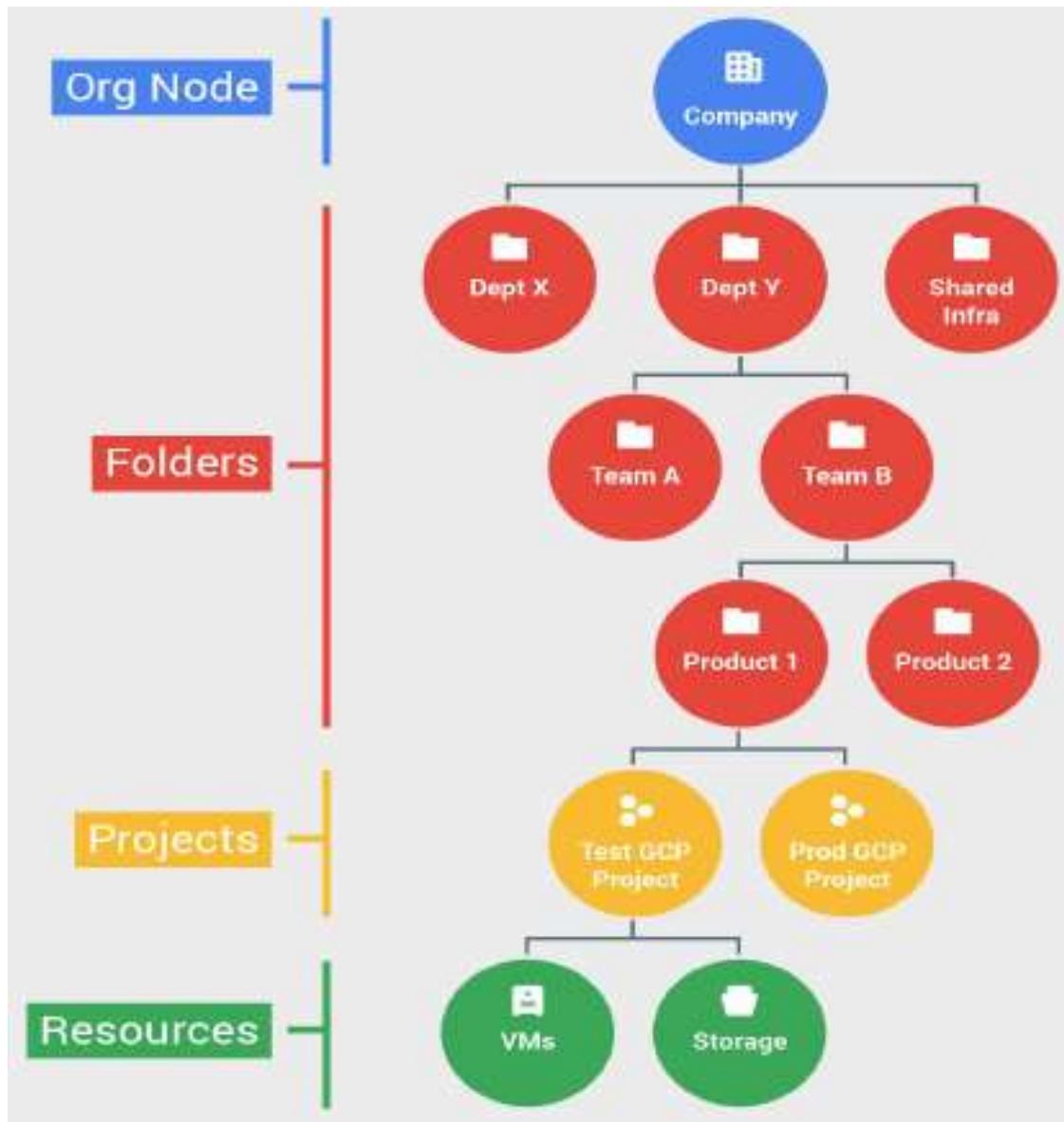
 Customer-managed    Google-managed

Responsibility	On-premises	Infrastructure as a Service	Platform as a Service	Managed services
Content				
Access policies				
Usage				
Deployment				
Web application security				
Identity				
Operations				
Access and authentication				
Network security				
OS, data, and content				
Audit logging				
Network				
Storage and encryption				
Hardware				

# 구글클라우드 자원 계층체계 (Resource Hierarchy)

자원의 계층구조 수준(Resource hierarchy levels)이 신뢰 경계선(trust boundary)을 결정

- 조직도에 따라 자원을 그룹화
- 계층 구조의 수준은 자원의 고립(isolation)과 신뢰 경계선(trust boundary)을 제공
- 상위 수준의 정책을 자동적으로 승계



# 구글클라우드 프로젝트 (Project)

모든 GCP 서비스는 프로젝트와 연계되어서 비용을 처리

- 자원 추적 및 가능 사용량(Quota) 할당
- 사용료 정산 (Billing)
- 권한(Permission) 및 인증서(Credential) 관리
- 서비스 사용 권한 및 API 사용 권한 설정

프로젝트는 아래와 같은 3가지 자격증명 속성(Identity Attribute)을 보유

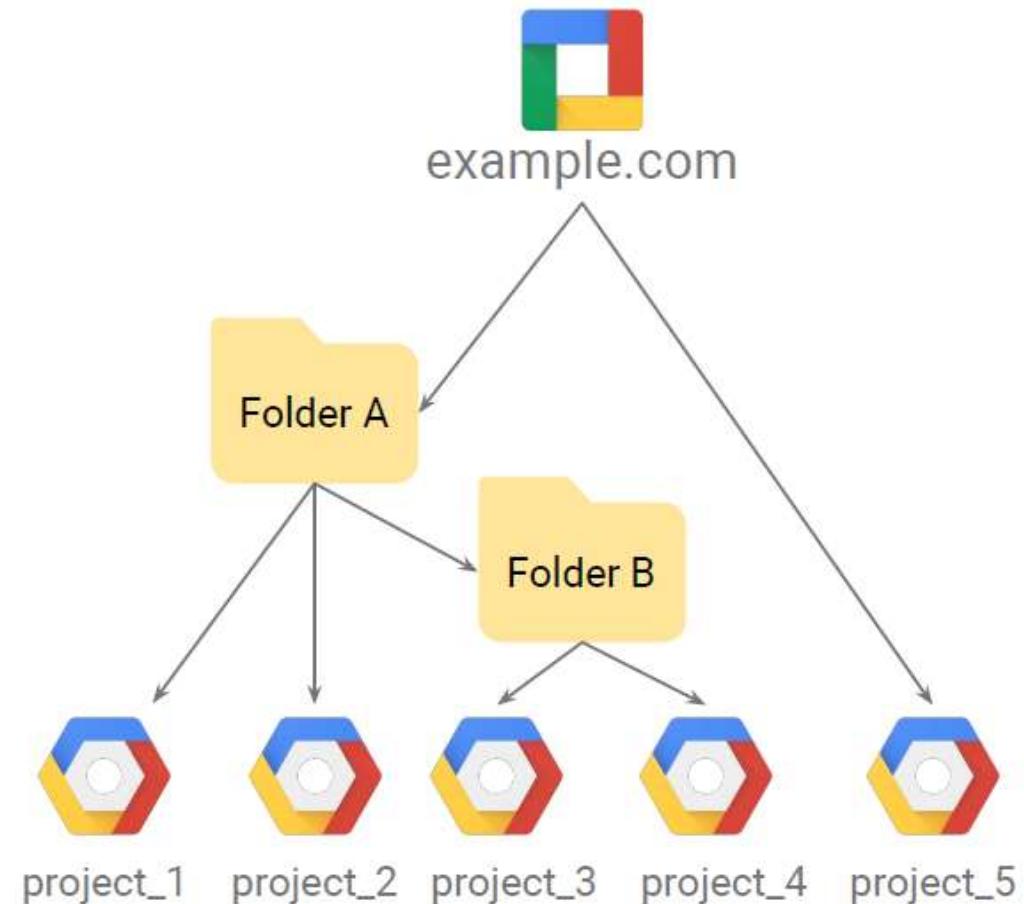
<b>Project ID</b>	Globally unique	Chosen by you	Immutable
<b>Project name</b>	Need not be unique	Chosen by you	Mutable
<b>Project number</b>	Globally unique	Assigned by GCP	Immutable

# 구글클라우드 폴더 (Folder)

## 폴더가 있어서 유연한 관리가 용이

- 조직도에 따른 프로젝트 그룹화 관리 용이
- 폴더는 프로젝트나 하위 폴더 등을 포함할 수 있음
- 폴더를 사용하여 정책(Policy) 설정 및 할당이 가능

→ 대기업 등에서 조직에 따른 권한 및 보안정책 관리 등이 쉽고 편해짐



# 구글클라우드 자원 계층체계와 IAM 역할 (Role)

An example IAM resource hierarchy

- A policy is set on a resource.
- Each policy contains a set of roles and role members.
- Resources inherit policies from parent.
- Resource policies are a union of parent and resource.
- A less restrictive parent policy overrides a more restrictive resource policy.



Who



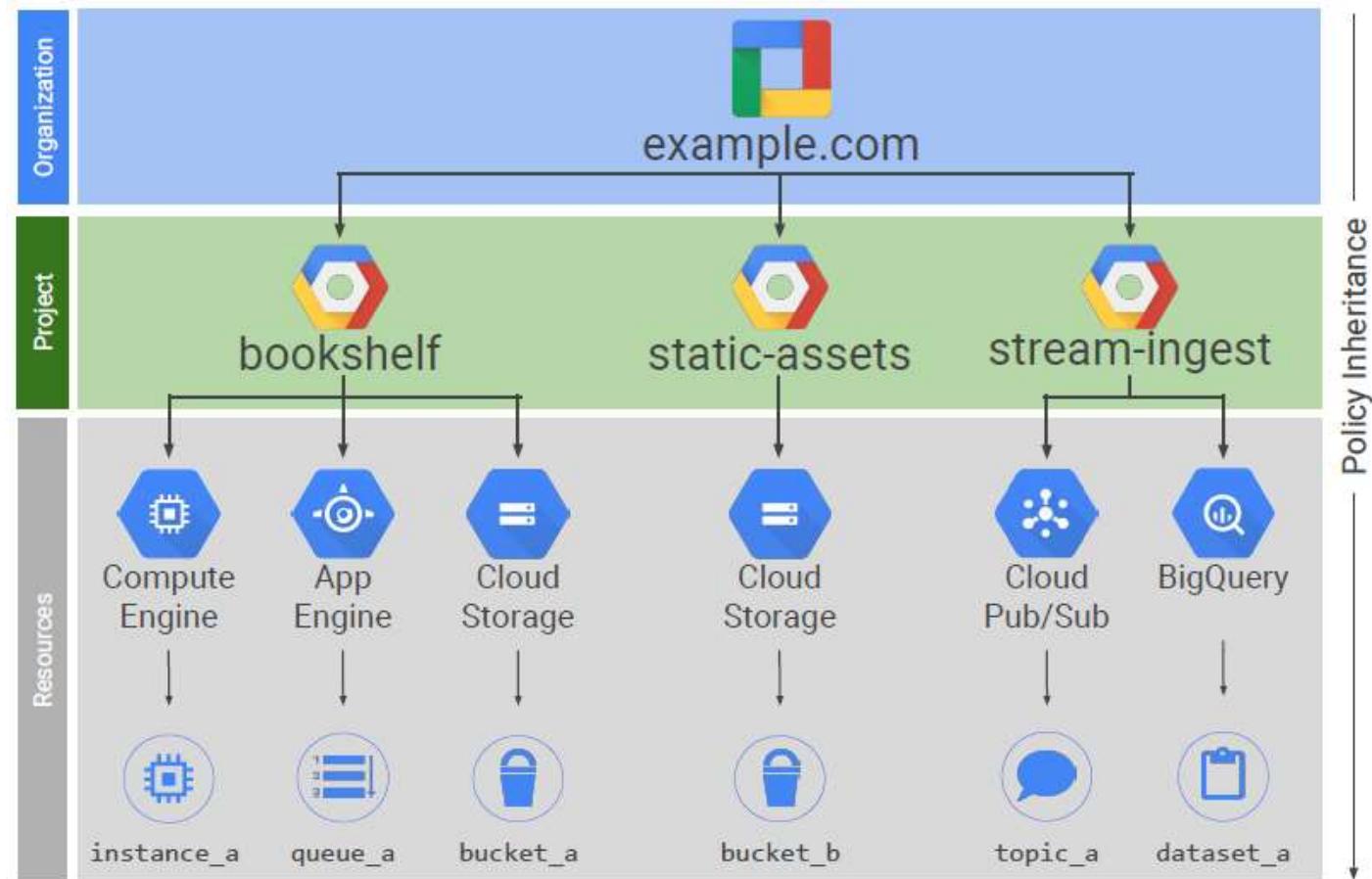
can do what



on which resource

account  
User identity

IAM roles

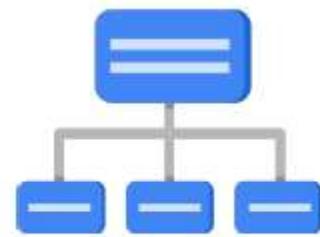


# 구글클라우드 IAM 소개

## Cloud IAM Objects



Cloud IAM



Organization



Folders



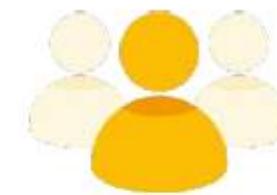
Projects



Resources



Roles



Members

# 구글클라우드 IAM 역할 : 기본 역할 (Primitive Roles)

## IAM Primitive Roles



Owner

- Invite members
- Remove members
- Delete projects
- And...



Editor

- Deploy applications
- Modify code
- Configure services
- And...



Viewer

- Read-only access



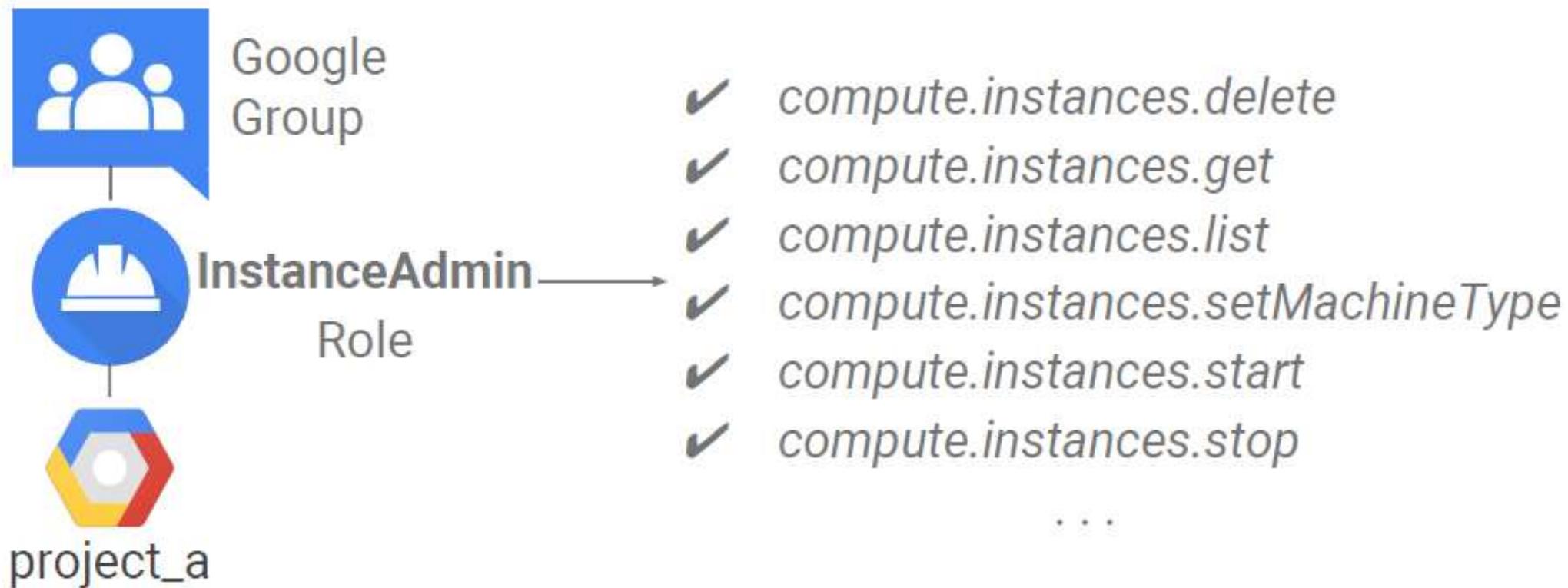
Billing  
administrator

- Manage billing
- Add and remove administrators

A project can have multiple owners, editors, viewers, and billing administrators.

# 구글클라우드 IAM 역할 : 사전 정의된 역할 (Predefined Roles)

- IAM predefined roles offer more fine-grained permissions on particular services



# 구글클라우드 IAM 역할 : 맞춤 역할 (Custom Roles)

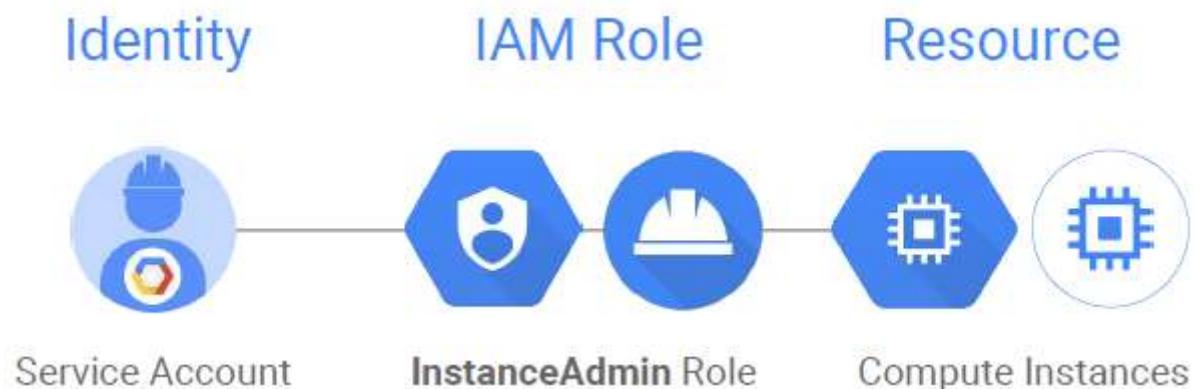
- IAM custom roles let you define a precise set of permissions



# 서비스 계정의 IAM

## 서비스 계정과 IAM

- Service accounts authenticate using keys.
- Google manages keys for Compute Engine and App Engine.
- You can assign a predefined or custom IAM role to the service account.



# 클라우드 아이덴티티

- **Cloud Identity**

Microsoft Active  
Directory or LDAP

Users and groups in  
your existing directory  
service

Google Cloud  
Directory Sync

Scheduled  
one-way sync



Users and groups in  
your Cloud Identity  
domain

# 구글클라우드와 상호작용하는 방법

- Interacting with GCP

## Cloud Platform Console

Web user  
interface



## Cloud Shell and Cloud SDK

Command-line  
interface



## Cloud Console Mobile App

For iOS and  
Android



## REST-based API

For custom  
applications



# 실습 : 구글클라우드 시작하기 (1)

- Gmail account를 이용하여 계정 만들기

The screenshot shows the Google Cloud Start page. At the top, there is a navigation bar with the Google Cloud logo, a search bar, and links for '문서', '지원', 'Language' (set to Korean), and '로그인'. Below the navigation bar, there is a large graphic featuring a stylized envelope icon, a white circle, and a green circle. The main heading '시작하기 전에 필요한 모든 사항' is displayed prominently. On the left, there is a section for 'Google Cloud Platform' with a logo, a brief description about building with \$300 free credit, and a blue button labeled 'GCP 무료로 사용해 보기'. To the right, there is a section titled 'GCP 리소스:' with links to '무료 등급', '문서', '빠른 시작', 'Marketplace', and '교육'.

Google Cloud

Google을 선택해야 하는 이유

솔루션

제품

가격 책정

시작하기

문서

지원

Language

로그인

영업팀에 문의

## 시작하기 전에 필요한 모든 사항

Google Cloud Platform

안전한 지능형 플랫폼으로 바로 빌드하세요. 신규 고객은 \$300 무료 크레딧을 사용하여 원하는 GCP 제품을 사용해 볼 수 있습니다.

GCP 무료로 사용해 보기

GCP 리소스:

- 무료 등급
- 문서
- 빠른 시작
- Marketplace
- 교육

<https://cloud.google.com/start/?hl=ko>

## 실습 : 구글클라우드 시작하기 (2)

- GCP Free-tier

# Google Cloud Platform 무료 등급

무료로 GCP를 배우고 빌드하세요.

[Console로 이동](#)



12개월

원하는 GCP 제품을 사용해 볼 수 있도록 \$300의 무료 크레딧이 제공됩니다.



항상 무료

자격 요건을 갖춘 고객에게는 무료 체험판 기간 동안과 종료 후에 해당 제품의 무료 사용량 한도를 제공합니다. 제공 서비스는 변경될 수 있습니다.

<https://cloud.google.com/free/?hl=ko>

## **Part II**

# **GCP 핵심 서비스 소개 및 실습**

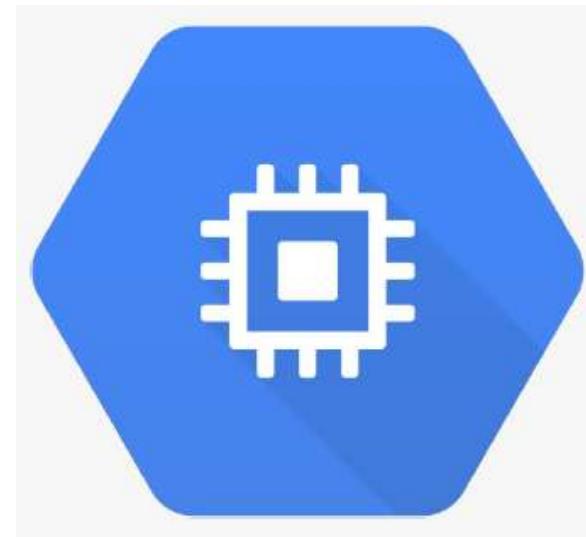
# **GCP Compute Resource**

## **- Compute Engine**

# GCP Compute Engine

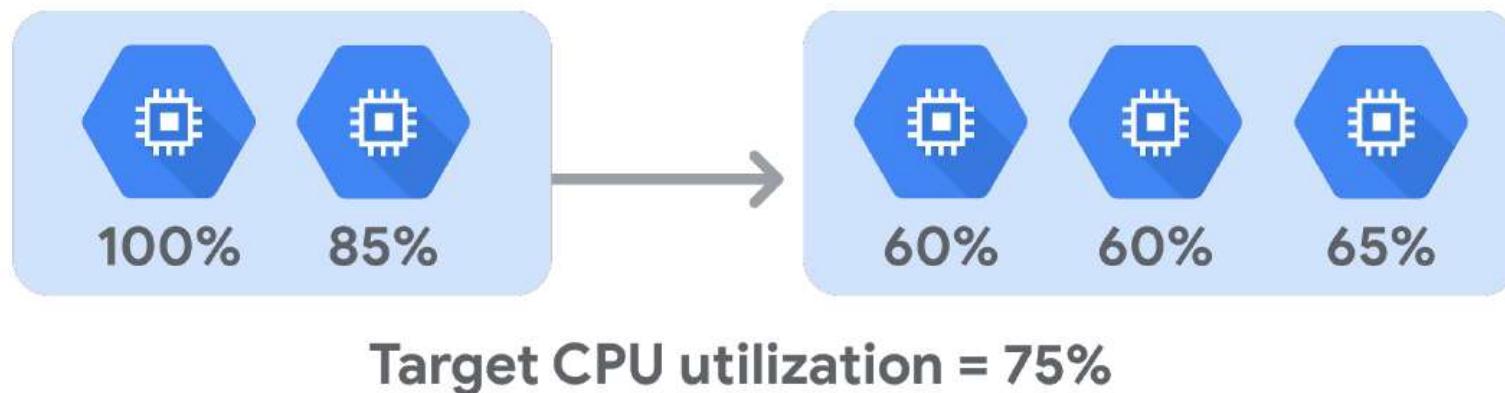
- Compute Engine 에서는 가상머신(Virtual Machine, VM)을 제공

- High CPU, high memory, standard and shared-core machine types
- Persistent disks
  - Standard, SSD, local SSD
  - Snapshots
- Resize disks with no downtime
- Instance metadata and startup scripts



# Auto-Scaling of VMs

- CPU 사용량 등 정해준 설정값에 따라 VM 숫자를 늘리고 줄임
  - Dynamically add/remove instances:
    - Increases in load
    - Decreases in load
  - Autoscaling policy:
    - CPU utilization
    - Load balancing capacity
    - Monitoring metrics
    - Queue-based workload



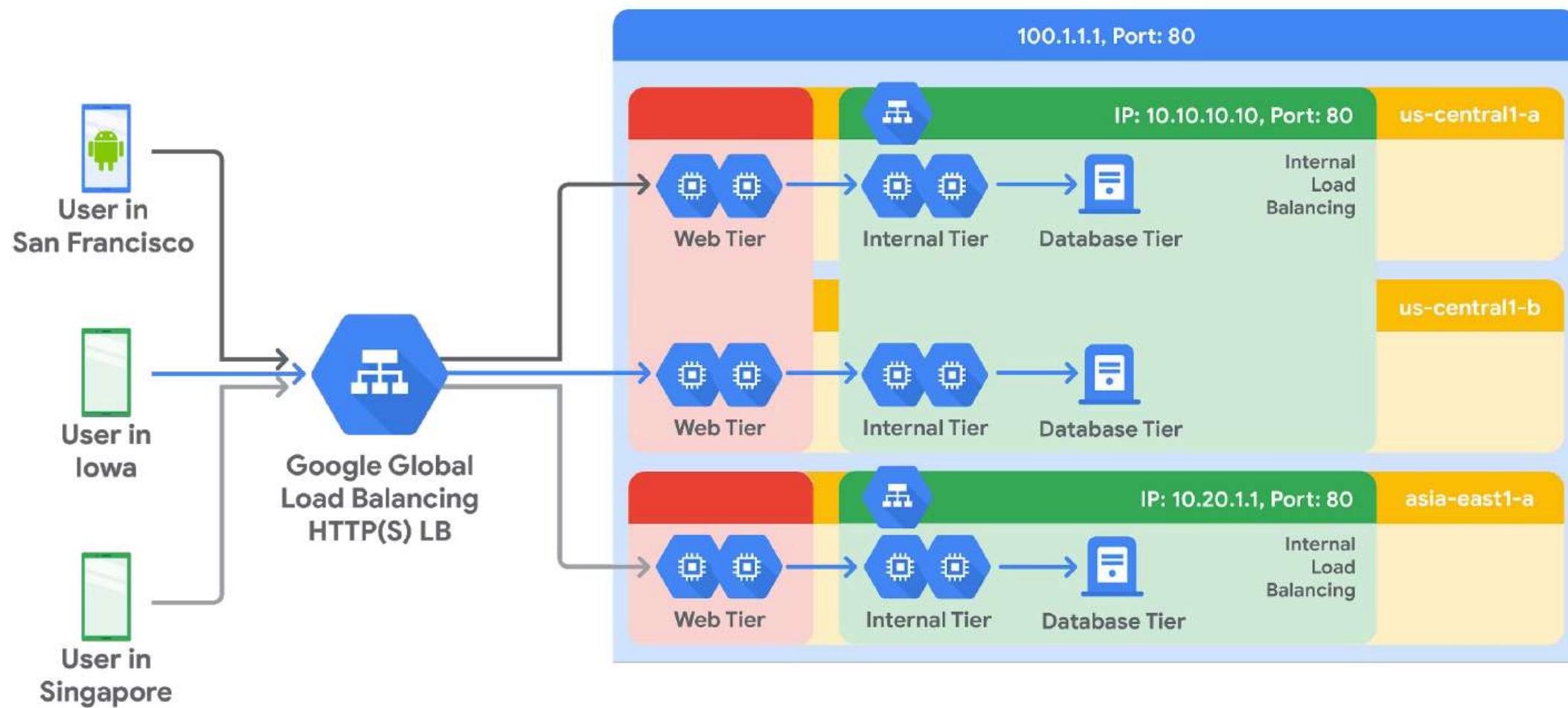
# GCP VPC Load Balancing Options

- 로드밸런서는 OSI Level 7 & 4 수준에서 제공

Global HTTP(S)	Global SSL Proxy	Global TCP Proxy	Regional	Regional internal
Layer 7 load balancing based on load	Layer 4 load balancing of non-HTTPS SSL traffic based on load	Layer 4 load balancing of non-SSL TCP traffic	Load balancing of any traffic (TCP, UDP)	Load balancing of traffic inside a VPC
Can route different URLs to different back ends	Supported on specific port numbers	Supported on specific port numbers	Supported on any port number	Use for the internal tiers of multi-tier applications

# Load-Balancing Example

- 일반적인 3-Tier 구조에서 Native Cloud Architecture 로드밸런싱 구조 예시



# **GCP Network Resource**

## **- Virtual Private Cloud**

# Google Virtual Private Cloud

## VPC objects 의 종류

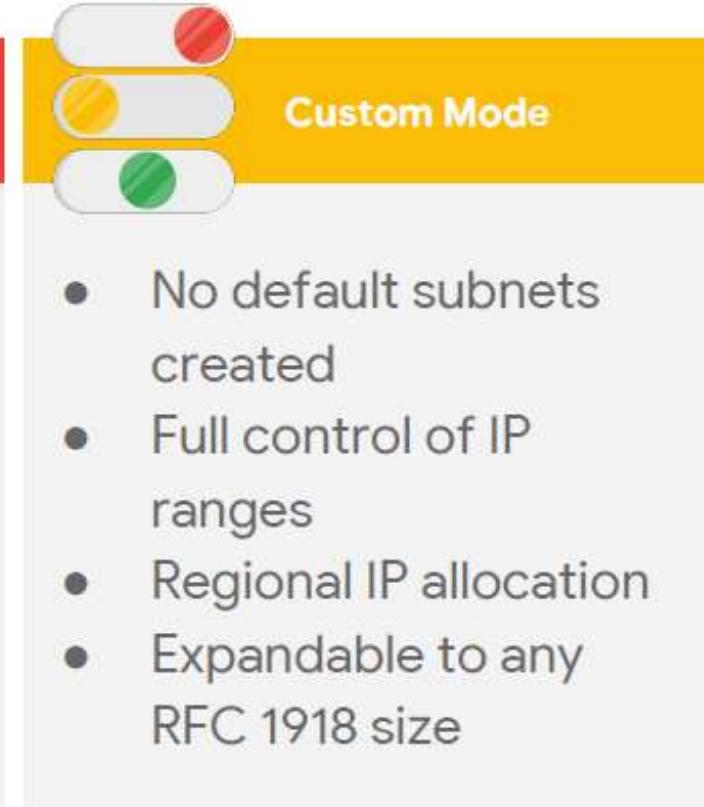
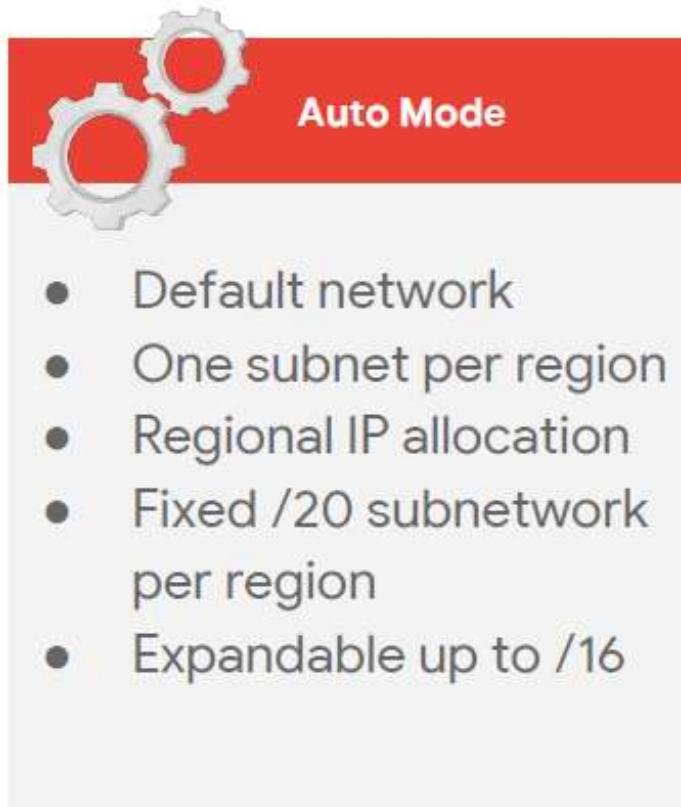
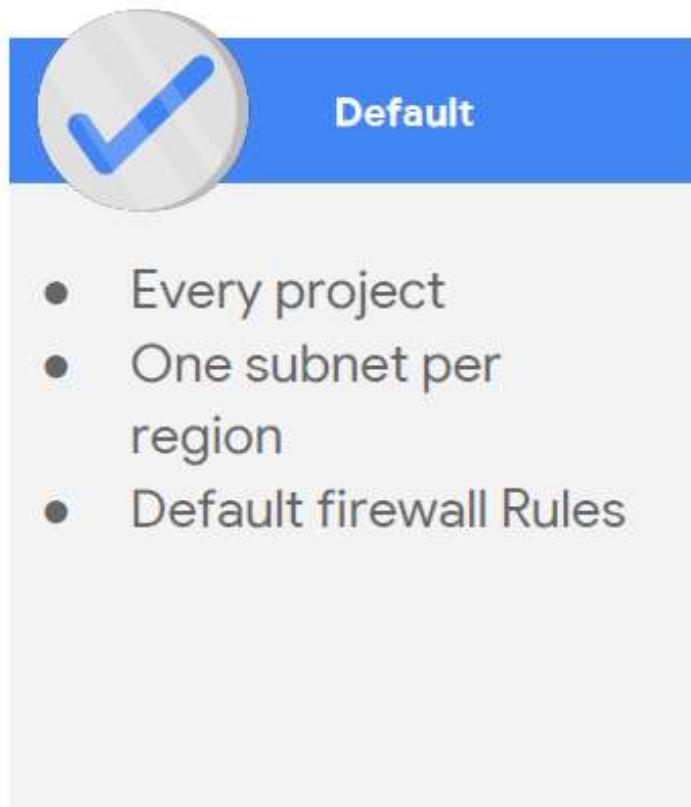


Virtual Private  
Cloud

- Projects
- Networks
  - Default, auto mode, custom mode
- Subnetworks
- Regions
- Zones
- IP addresses
  - Internal, external, range
- Virtual machines (VMs)
- Routes
- Firewall rules

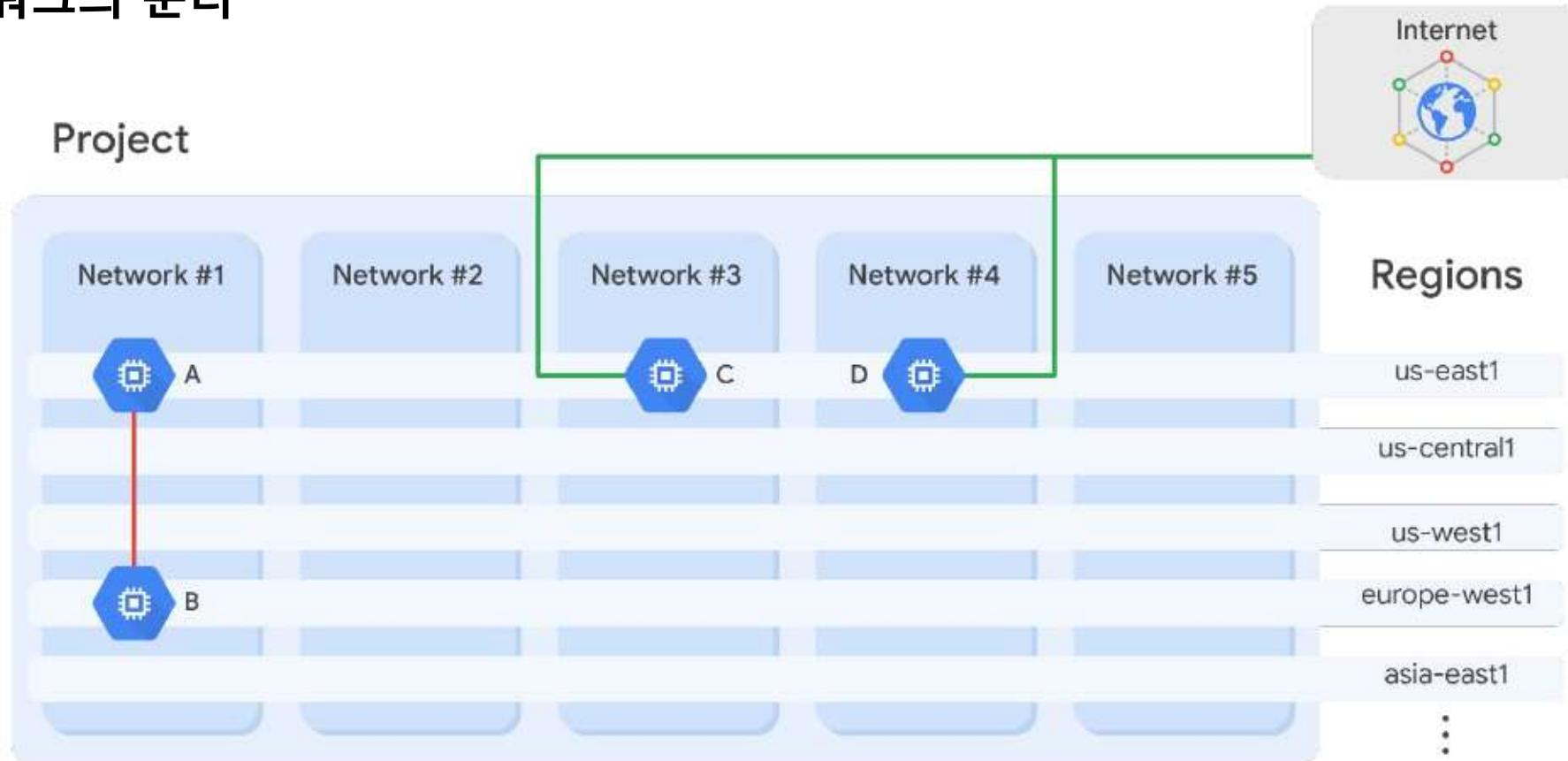
# Google Virtual Private Cloud

## 3가지 종류의 VPC



# Google Virtual Private Cloud

## VPC 네트워크의 분리

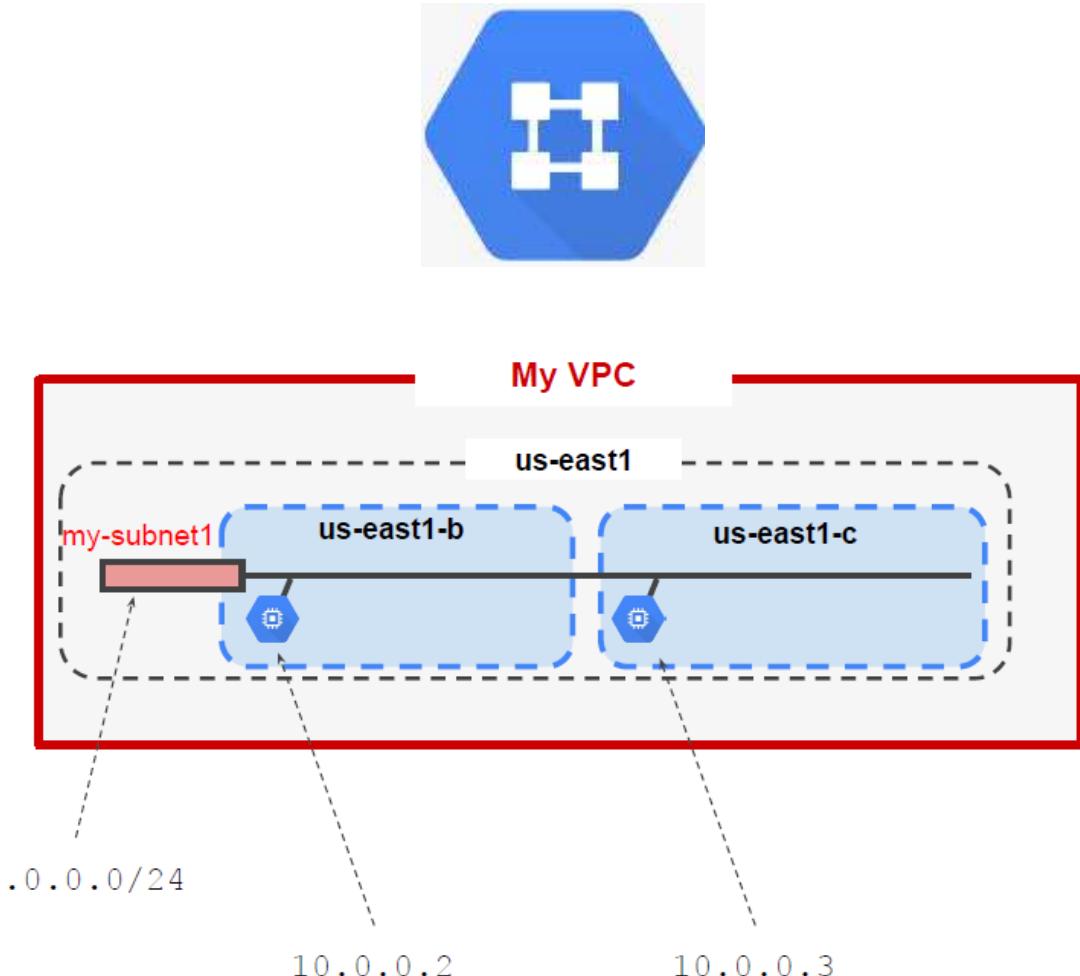


- A and B can communicate over internal IPs even though they are *in different regions*.
- C and D must communicate over external IPs even though they are *in the same region*.

# Google Virtual Private Cloud (VPC) Networking

- 프로젝트별로 가상 VPC 네트워크를 할당하고 관리

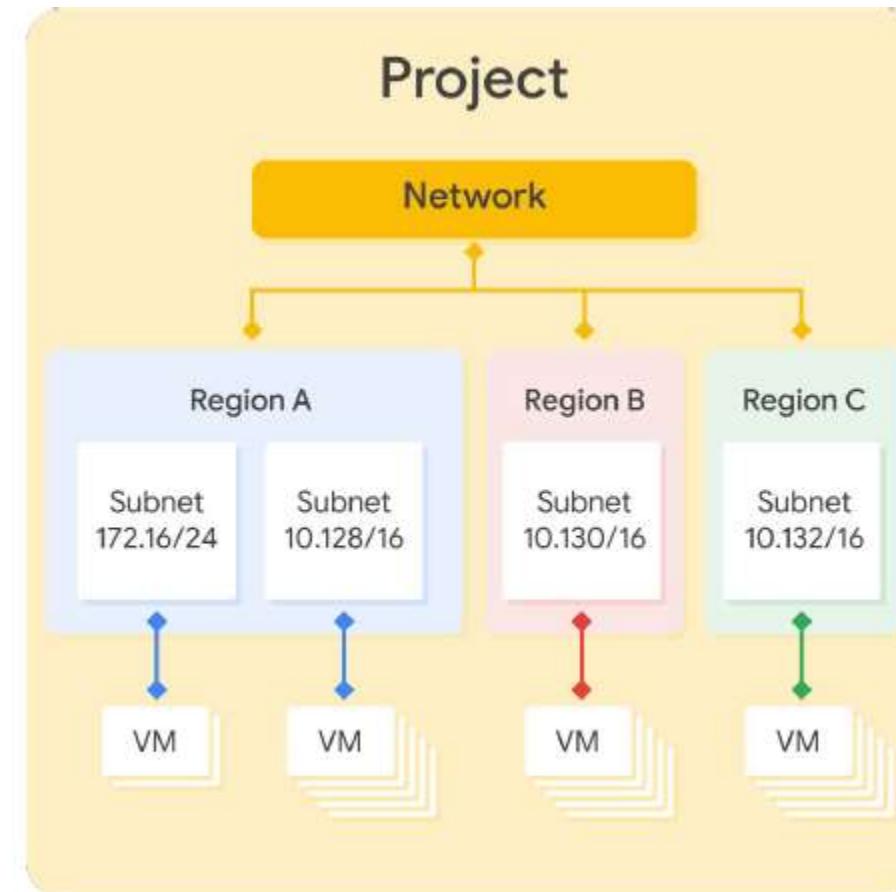
- Each VPC network is contained in a GCP project.
- You can provision Cloud Platform resources, connect them to each other, and isolate them from one another.
- Google Cloud VPC networks are global; subnets are regional



# Google Virtual Private Cloud

## Subnet의 확대

- Cannot overlap with other subnets
- Inside the RFC 1918 address spaces
- Can expand but not shrink
- Auto mode can be expanded from /20 to /16
- Avoid large subnets



# Google Virtual Private Cloud

## VM의 내부 IP 주소와 외부 IP 주소 할당



Cloud External  
IP Addresses



### Internal IP

Allocated from subnet range to VMs by DHCP

DHCP lease is renewed every 24 hours

VM name + IP is registered with network-scoped DNS

### External IP

Assigned from pool (ephemeral)

Reserved (static)

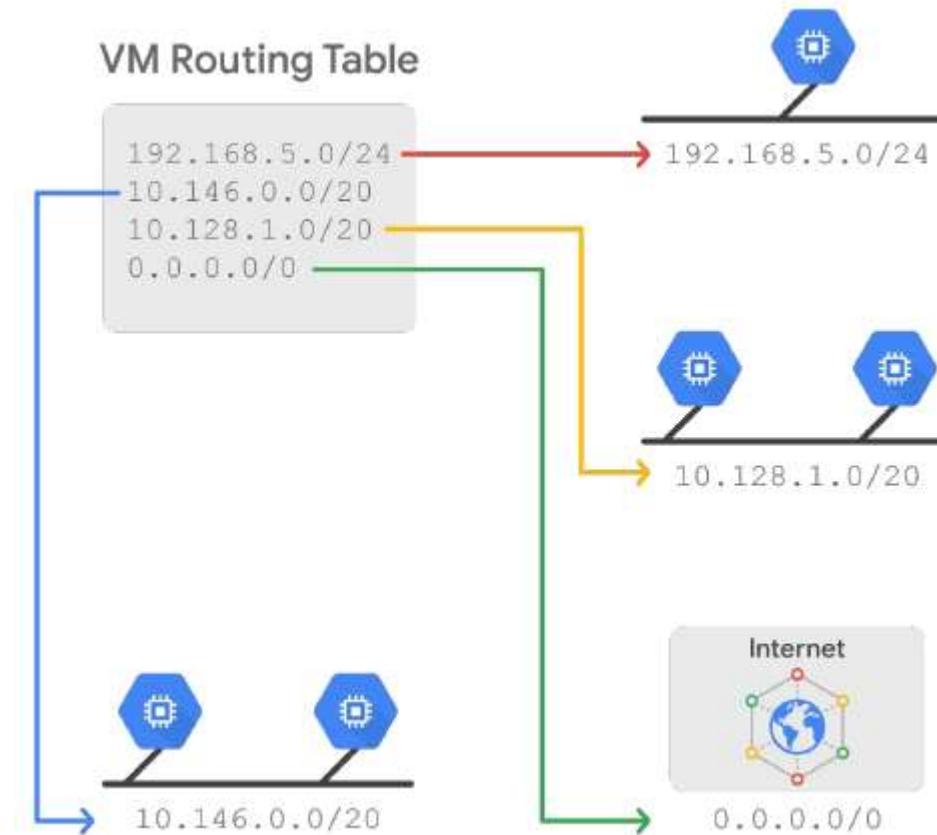
Billed when not attached to a running VM

VM doesn't know external IP; it is mapped to the internal IP

# Google Virtual Private Cloud

## 목적지 네트워크로 가는 IP주소 Routing Table

- Destination in CIDR notation
- Applies to traffic egressing a VM
- Forwards traffic to most specific route
- Traffic is delivered only if it also matches a firewall rule
- Created when a subnet is created
- Enables VMs on same network to communicate



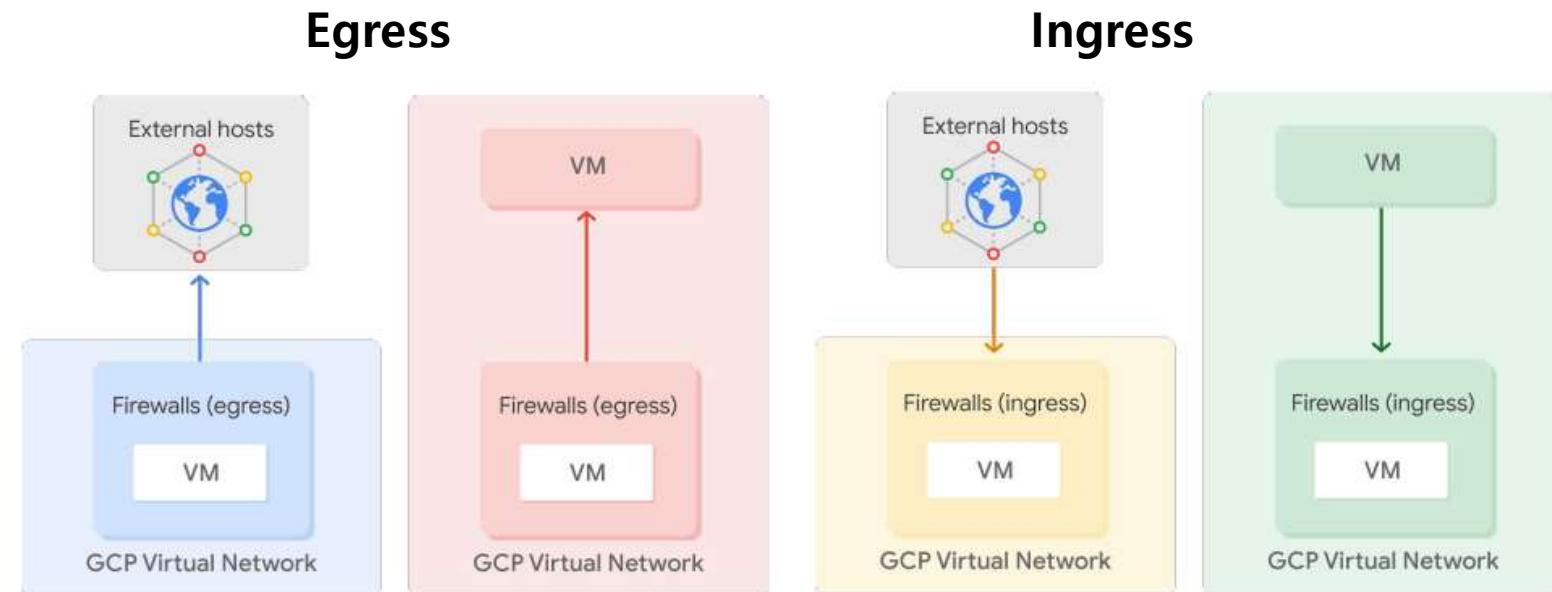
# Google Virtual Private Cloud

## 방화벽 (Firewall Rules)

- VPC network functions as a distributed firewall.
- Firewall rules are applied to the network as a whole.
- Connections are allowed or denied at the instance level.
- Firewall rules are stateful.
- Implied deny all ingress and allow all egress.



Cloud Firewall  
Rules

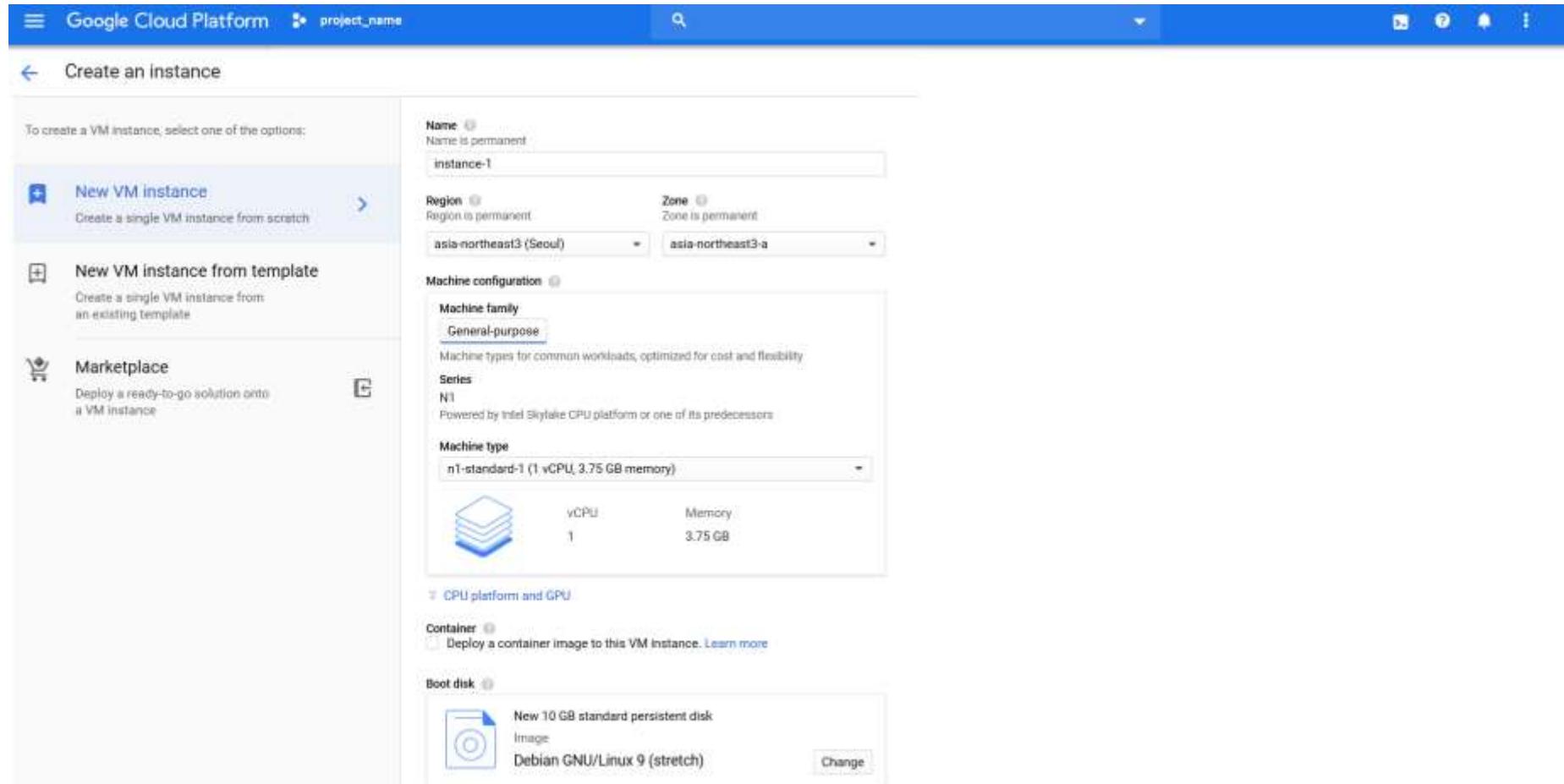


- CIDR ranges
- Protocols

- Ports
- Allow vs. Deny

# 실습 : VM Instance 생성 & VPC

- VM Instance를 각기 다른 region/subnet에 생성하고 서로 교신할 수 있도록 방화벽 설정하기

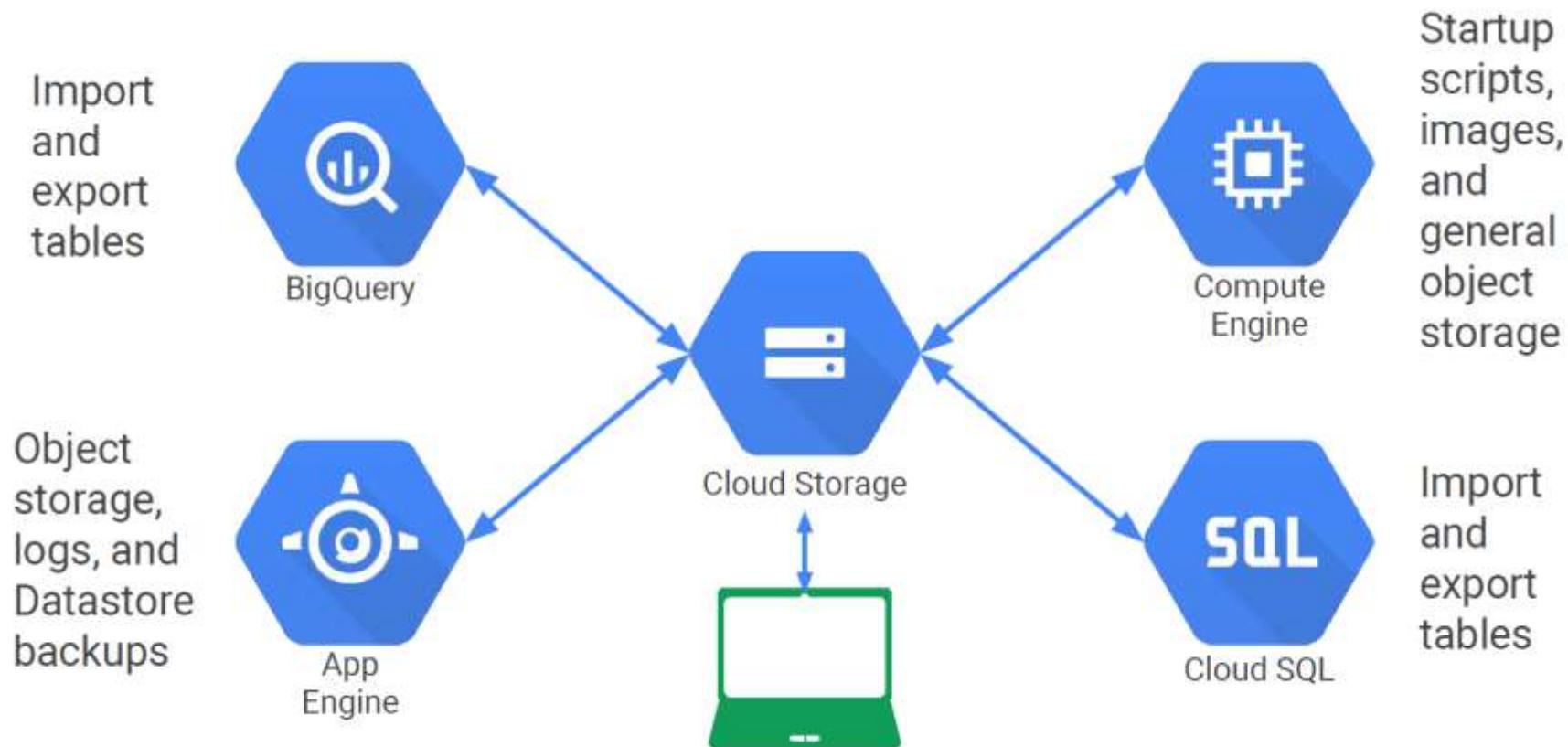


<https://cloud.google.com/compute/docs/instances/create-start-instance?hl=ko>

# GCP 스토리지

# GCP Cloud Storage 소개

- 바이너리 오브젝트 스토리지 (Binary large-object storage)
- 인터넷 스케일의 고성능 스토리지
- 완전관리형으로 용량 관리가 불필요
- 데이터저장이나 송수신 때 암호화 지원
- Online 과 offline 지원



# GCP Cloud Storage Bucket

- 스토리지 인스턴스(Instance)를 버킷(Bucket)이라고 부름
- 인터넷 스케일로 On/Off-Line을 모두 지원하기 때문에 Globally Unique한 이름이 필요  
→ 보통 프로젝트 이름을 활용하여 Naming
- 버킷 전체적인 IAM 과 ACL 관리가 가능하며 파일 오브젝트 개별로 ACL 권한 관리가 가능
- 수명주기(Lifecycle) 및 버전관리가 가능하여 데이터 백업이나 재해 복구용도로 사용도 가능

Bucket attributes	Bucket contents
Globally unique name	Files (in a flat namespace)
Storage class	
Location (region or multi-region)	
IAM policies or Access Control Lists	Access Control Lists
Object versioning setting	
Object lifecycle management rules	

# GCP Cloud Storage Class

- 자주 사용하는 파일은 접속 지역 Coverage에 따라 Regional vs. Multiregional Class 사용
- 백업이나 재해복구용으로는 접속 빈도에 따라 Nearline이나 Coldline을 사용  
→ Lifecycle Management 활용 가능

	Multi-regional	Regional	Nearline	Coldline
Intended for data that is...	Most frequently accessed	Accessed frequently within a region	Accessed less than once a month	Accessed less than once a year
Availability SLA	99.95%	99.90%	99.00%	99.00%
Access APIs	<i>Consistent APIs</i>			
Access time	<i>Millisecond access</i>			
<u>Storage price</u>	Price per GB stored per month			
<u>Retrieval price</u>	Total price per GB transferred			
Use cases	Content storage and delivery	In-region analytics, transcoding	Long-tail content, backups	Archiving, disaster recovery

# 실습 : Cloud Storage Bucket

- Bucket을 특정 region에 생성하고 파일을 저장하고 다시 download 하기

The screenshot shows the 'Create a bucket' wizard in the Google Cloud Platform. On the left, a sidebar lists 'Storage', 'Browser', 'Transfer', 'Transfer for on-premises', 'Transfer Appliance', and 'Settings'. The main area has a title 'Create a bucket' with a back arrow. It contains several steps:

- Name your bucket**: A text input field with placeholder 'Ex. 'example', 'example\_bucket-1', or 'example.com'' and a red border. Below it is a 'Required' label.
- CONTINUE** button
- Choose where to store your data**
- Choose a default storage class for your data**
- Choose how to control access to objects**
- Advanced settings (optional)**

At the bottom are 'CREATE' and 'CANCEL' buttons. To the right, there's a 'Monthly cost estimate' section with fields for 'Storage size' (GB), 'Data retrieval size' (GB), and 'Operations' (per-month) for Class A and Class B operations.

<https://cloud.google.com/storage/docs/creating-buckets?hl=ko>

# GCP 데이터베이스

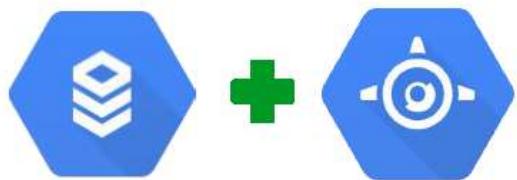
# GCP Cloud SQL 소개

- Cloud SQL은 관리형 Database
- 기본 엔진으로 MySQL(MariaDB), PostgreSQL, SQL Server를 제공
  - Offers MySQL and PostgreSQL databases as a service
  - Automatic replication
  - Managed backups
  - Vertical scaling (read and write)
  - Horizontal scaling (read)
  - Google security



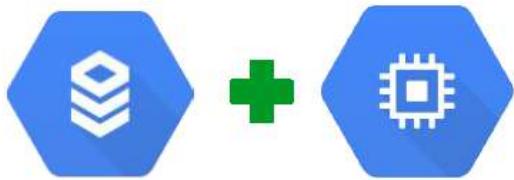
<https://cloud.google.com/sql/?hl=ko>  
<https://cloud.google.com/sql/docs/?hl=ko>

# GCP Cloud SQL : 다른 GCP 서비스와의 연계 활용



Cloud SQL can be used with App Engine using standard drivers.

You can configure a Cloud SQL instance to follow an App Engine application.



Compute Engine instances can be authorized to access Cloud SQL instances using an external IP address.

Cloud SQL instances can be configured with a preferred zone.



Cloud SQL can be used with external applications and clients.

Standard tools can be used to administer databases.

External read replicas can be configured.

# GCP 스토리지 비교

- 완전 불규칙한 비정형 정보 저장 → 파일 형태로 Cloud Storage 활용
- 규칙적 정형 데이터이며 실시간 중요 → Cloud SQL (혹은 글로벌 scale 은 Cloud Spanner)
- 다소 규칙적 NoSQL 데이터 → Cloud Datastore / Bigtable
- Data Warehouse → BigQuery

	Cloud Datastore	Bigtable	Cloud Storage	Cloud SQL	Cloud Spanner	BigQuery
Type	NoSQL document	NoSQL wide column	Blobstore	Relational SQL for OLTP	Relational SQL for OLTP	Relational SQL for OLAP
Transactions	Yes	Single-row	No	Yes	Yes	No
Complex queries	No	No	No	Yes	Yes	Yes
Capacity	Terabytes+	Petabytes+	Petabytes+	Up to ~10 TB	Petabytes	Petabytes+
Unit size	1 MB/entity	~10 MB/cell ~100 MB/row	5 TB/object	Determined by DB engine	10,240 MiB/ row	10 MB/row

# 실습 : Cloud SQL

- 2세대 MySQL 인스턴스 만들기

SQL [Create an instance](#)

Choose your database engine

**MySQL**  
Versions: 5.6, 5.7  
→ Choose MySQL

**PostgreSQL**  
Versions: 9.6, 11  
→ Choose PostgreSQL

**SQL Server BETA**  
Versions: 2017  
→ Choose SQL Server

First Generation MySQL instances are being decommissioned soon, but you can create one [here](#).

Want more context on the Cloud SQL database engines? [Learn more](#)

<https://cloud.google.com/sql/docs/mysql/create-instance?hl=ko>

# **GCP 관리**

# GCP 클라우드 자원 관리

- GCP는 자원관리 가시성이 뛰어남

The screenshot shows the left sidebar of the Google Cloud Platform interface. At the top is the 'Google Cloud Platform' logo. Below it are several navigation items: Home, BigQuery, Dataflow, and a 'PRODUCTS' section which includes Marketplace, Billing, APIs & Services, Support, IAM & admin, Getting started, and Security. The 'IAM & admin' item has a dropdown menu open, showing options like IAM, Identity & Organization, Policy Troubleshooter, Organization policies, Quotas, Service accounts, Labels, Settings, Privacy & Security, Cryptographic keys, Identity-Aware Proxy, Roles, Audit Logs, and Manage resources. The 'Manage resources' option is highlighted with a light gray background.

The screenshot shows the 'Manage resources' page. At the top, there are buttons for 'CREATE PROJECT' and 'DELETE'. Below is a 'Filter tree' button. The main area is a table with columns: Name, ID, Status, Charges, Labels, and Actions. There are four rows of data:

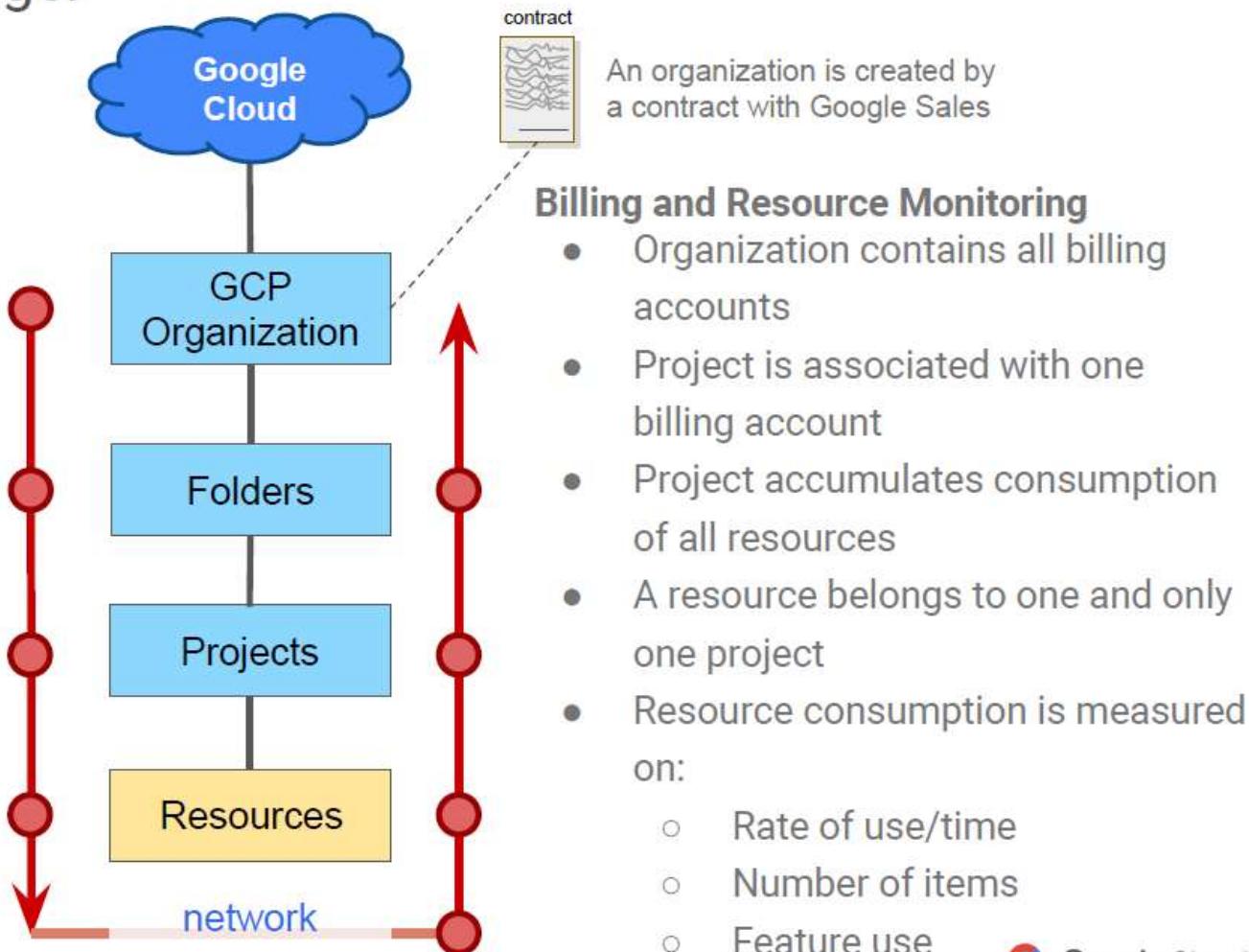
Name	ID	Status	Charges	Labels	Actions
No organization	0				⋮
fintech-info	fintech-info				⋮
ml-test-191114	ml-test-191114				⋮
test20191004	test20191004				⋮

Below the table, the text '0 RESOURCES PENDING DELETION' is displayed.

# GCP 클라우드 자원 관리

- 조직 / IAM / 자원관리 / 비용청구 등이 모두 일관성 있게 연계되어 있음

## Cloud Resource Manager



# GCP 클라우드 Quota

Google Cloud Platform fintech-info

Quotas + EDIT QUOTAS

Quota type	Service	Metric	Location	Current Usage	7 Day Peak Usage	Limit
All quotas	All services	All metrics	All locations	15.519	171	2,000
<input type="checkbox"/> Service		Location				
<input type="checkbox"/> Compute Engine API	Global	15.519	171	2,000		
List requests per 100 seconds						
<input type="checkbox"/> Cloud Runtime Configuration API	Global	0 B	446 B	4,194,304 B (4.194 MB)		
Config Storage Usage						
<input type="checkbox"/> Stackdriver Logging API	Global	0.001	1	60,000		
Log ingestion requests per minute						
<input type="checkbox"/> Compute Engine API	Global	0.006	0.006	2,000		
Queries per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	0.005	0.005	2,000		
Read requests per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	3	3	Unlimited		
Queries per day						
<input type="checkbox"/> Compute Engine API	Global	0	0	2,000		
Operation read requests per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	0	0	1,000		
Heavy-weight read requests per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	0	0	1,000		
Heavy-weight mutation requests per 100 seconds						

# GCP 클라우드 예산 및 알람 설정

Google Cloud Platform ≡ 🔍 ▼

Billing ← Create Budget

- Overview
- Reports
- Cost table
- Cost breakdown
- Commitments
- Budgets & alerts**
- Billing export
- Transactions
- Payment settings
- Payment method
- Account management

1 Scope — 2 Amount — 3 Actions

A budget enables you to track your actual spend against your planned spend.

Name \*

A budget can be scoped to focus on a specific set of resources.

Projects  ▾

Products  ▾

**NEXT** CANCEL

# GCP 클라우드 비용계산기

The screenshot shows the Google Cloud Platform Pricing Calculator. At the top, there is a navigation bar with the Google Cloud logo, a search icon, and language selection (Language: ko). Below the navigation bar, the title "Google Cloud Platform Pricing Calculator" is displayed, along with a note that prices are up to date as of January 31, 2020. A horizontal menu bar below the title contains icons for various services: COMPUTE ENGINE, APP ENGINE, KUBERNETES ENGINE, CLOUD RUN, CLOUD STORAGE, NETWORKING, BIGQUERY, BIGQUERY ML, and DATA. An "Estimate" button is located on the right side of this menu. Below the menu, there is a search bar with the placeholder text "Search for a product you are interested in," followed by a section titled "Instances". This section includes fields for "Number of instances" (with a dropdown arrow and a question mark icon), "Operating System / Software" (set to "Free: Debian, CentOS, CoreOS, Ubuntu, or other User Provided OS" with a dropdown arrow and a question mark icon), "Machine Class" (set to "Regular" with a dropdown arrow and a question mark icon), "Machine Family" (set to "General purpose" with a dropdown arrow and a question mark icon), and "Series" (set to "N1" with a dropdown arrow and a question mark icon).

<https://cloud.google.com/sql/docs/mysql/create-instance?hl=ko>

# 각 업체별 전략 – Google GCP

- 멀티클라우드 모니터링을 완벽 지원. 3<sup>rd</sup> Party Solution을 제외하고는 사실상 유일.

## Stackdriver

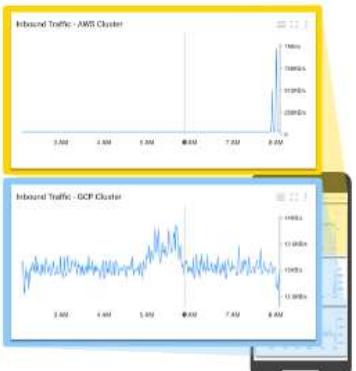
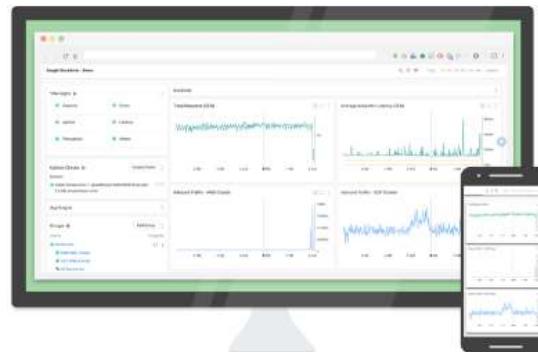
서비스, 컨테이너, 애플리케이션, 인프라를 모니터링 및 관리합니다.

[무료로 사용해 보기](#)

[이 제품의 문서 보기](#)

### 코드 및 애플리케이션을 완벽하게 관찰

Stackdriver는 측정항목, 로그, 이벤트를 인프라에서 집계하고 개발자와 운영자에게 관측 가능한 신호를 다양하게 제공하여 근본 원인 분석의 속도를 높이고 평균 문제 해결 시간(MTTR)을 단축합니다. Stackdriver는 광범위한 통합 또는 여러 개의 '관리 창'이 필요하지 않으며 개발자가 특정 클라우드 제공업체만 사용하도록 강제하지 않습니다.



### 여러 클라우드 및 온프레미스 인프라에서 작동

Stackdriver는 처음부터 클라우드 기반 애플리케이션을 위해 구축되었습니다. Google Cloud Platform, Amazon Web Services, 온프레미스 인프라, 하이브리드 클라우드 등, 실행 환경에 관계없이 모든 클라우드 계정 및 프로젝트의 측정항목, 로그, 메타데이터를 모아 포괄적인 단일 뷰에서 환경을 확인할 수 있으므로 서비스 동작을 빠르게 이해하고 조치를 취할 수 있습니다.

# **Part III**

# **Google Kubernetes Engine (GKE)**

# Kubernetes Engine 개요

# K8s provides container-centric infrastructure

Once specific containers are no longer bound to specific machines/VMs,  
host-centric infrastructure no longer works

- **Scheduling:** Decide where my containers should run
- **Lifecycle and health:** Keep my containers running despite failures
- **Scaling:** Make sets of containers bigger or smaller
- **Naming and discovery:** Find where my containers are now
- **Load balancing:** Distribute traffic across a set of containers
- **Storage volumes:** Provide data to containers
- **Logging and monitoring:** Track what's happening with my containers
- **Debugging and introspection:** Enter or attach to containers
- **Identity and authorization:** Control who can do things to my containers

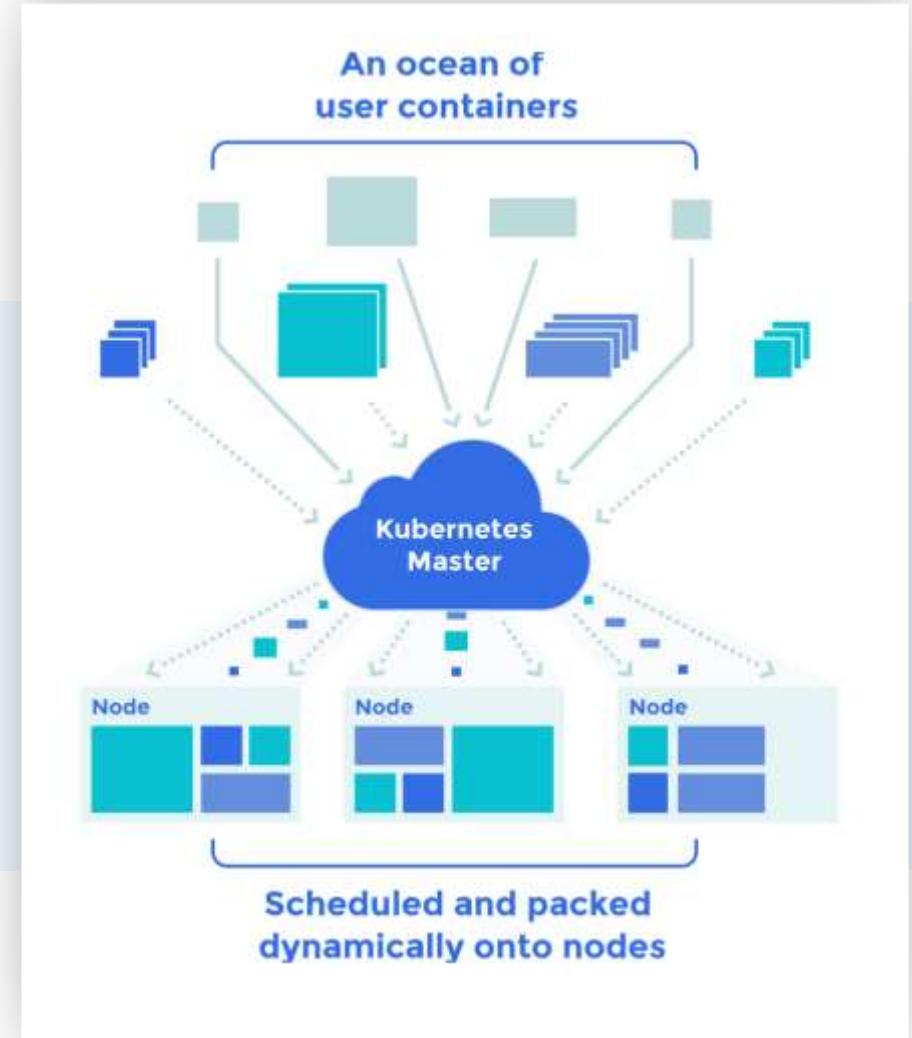


# In simple terms...

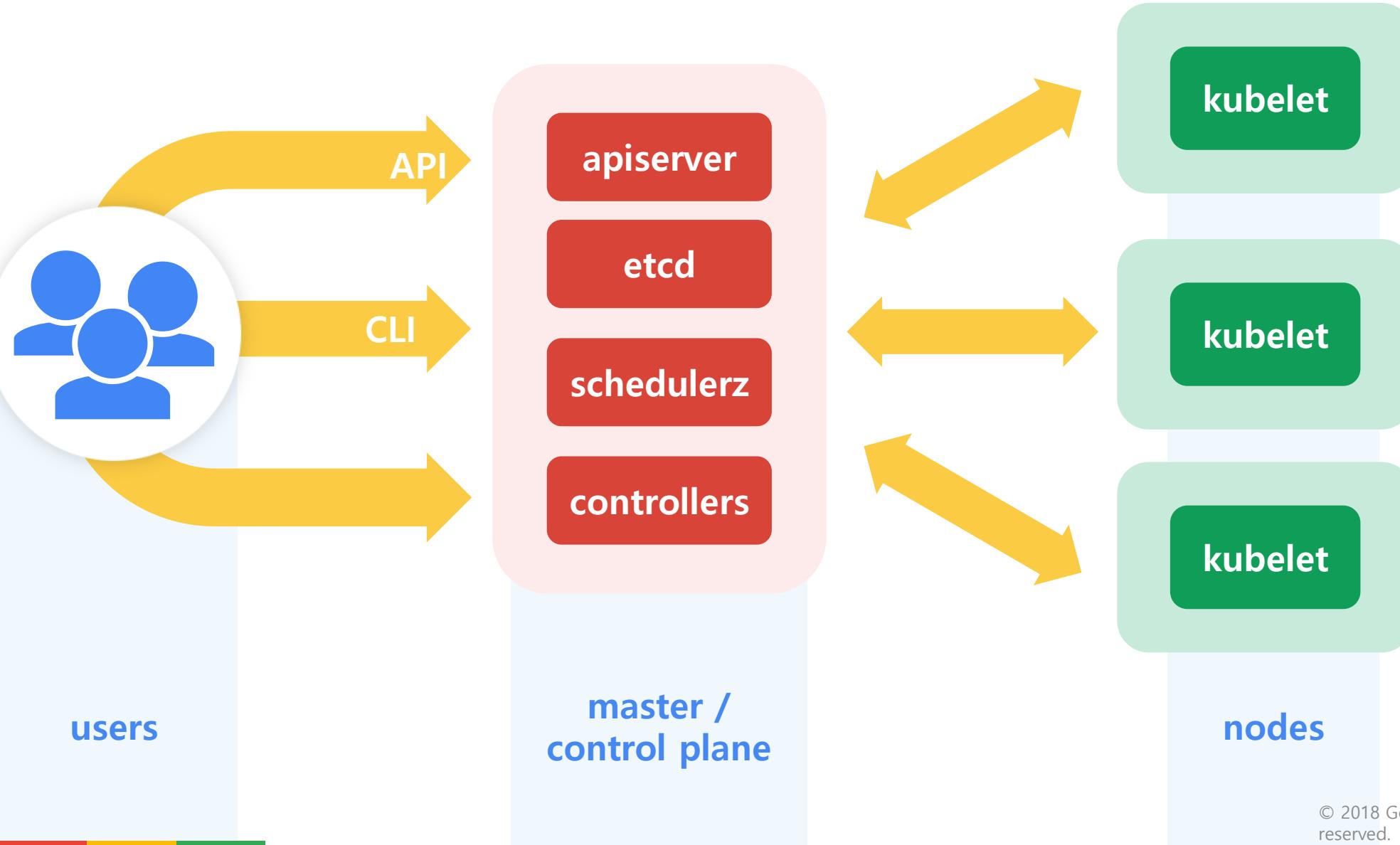


It provides features similar to an OS for a host:

- Scheduling workload
- Finding the right host to fit your workload
- Monitoring health of the workload
- Scaling it up and down as needed
- Moving it around as needed



# The 10,000-foot view



# Control loops

- Drive **current state** → **desired state**
- Act independently
- APIs — no **shortcuts** or back doors
- Observed state is truth
- Recurring pattern in the system



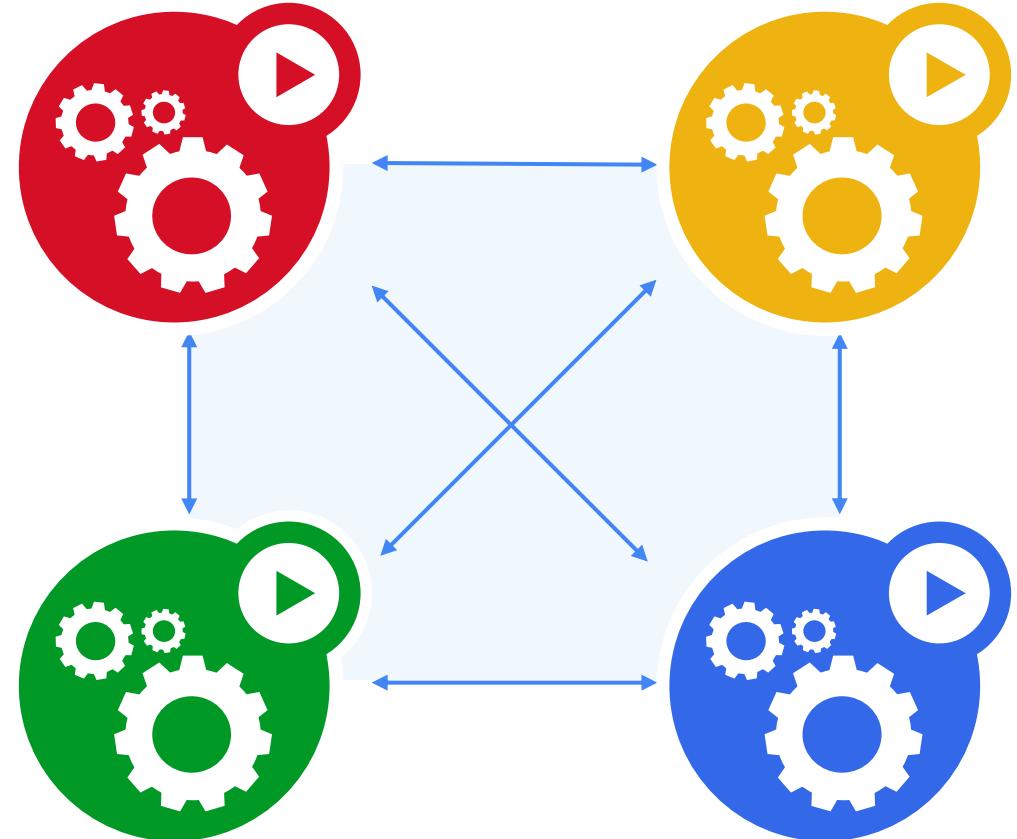
Examples: ReplicationController,  
IngressController

# **Kubernetes Networking**

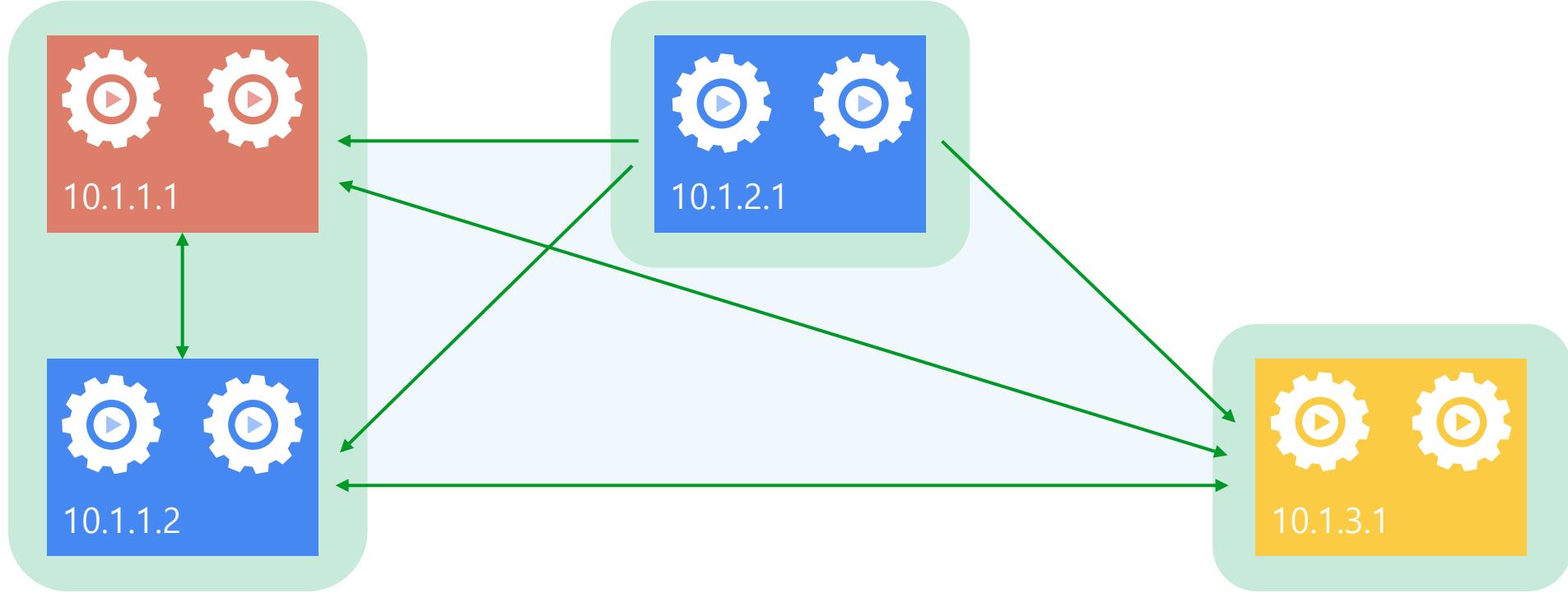
## **Design Principles**

# Networking in Kubernetes

- ▶ Pods get unique IPs
  - IPs are cluster-scoped
- ▶ Pods can reach each other directly
  - Even across nodes
- ▶ **No brokering** of port numbers
  - Each pod can use any port on its IP address
- ▶ **This is a fundamental requirement**
  - Can be L3 routed
  - Can be underlaid (cloud)
  - Can be overlaid (SDN)
  - Kubernetes Engine uses routes and alias IPs



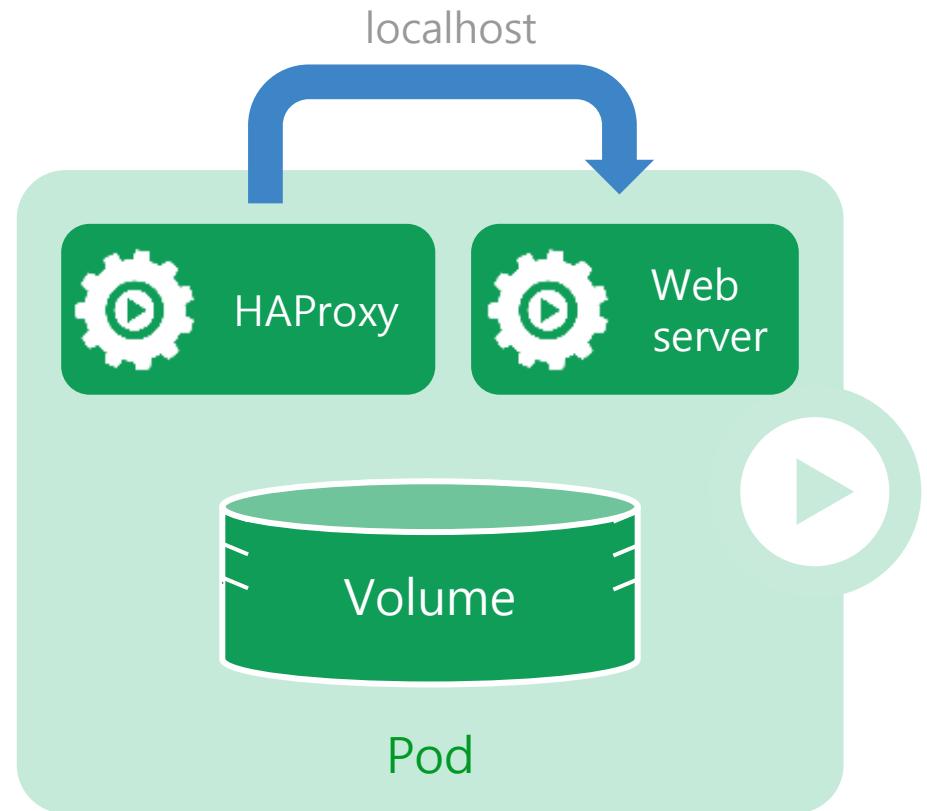
# Networking in Kubernetes



# In simple terms...

This goes well with the following analogy...

- Pods as VMs of the container world
- Containers in a pod as processes in a VM
- Volumes as disks on a VM
- VMs have individual IPs → So do pods
- VMs run processes → Pods run containers
- Processes in a VM
  - Share the host's IP address
  - Need to coordinate port allocation
  - Share the disk



# **Kubernetes Networking**

## **Labels and Selectors**

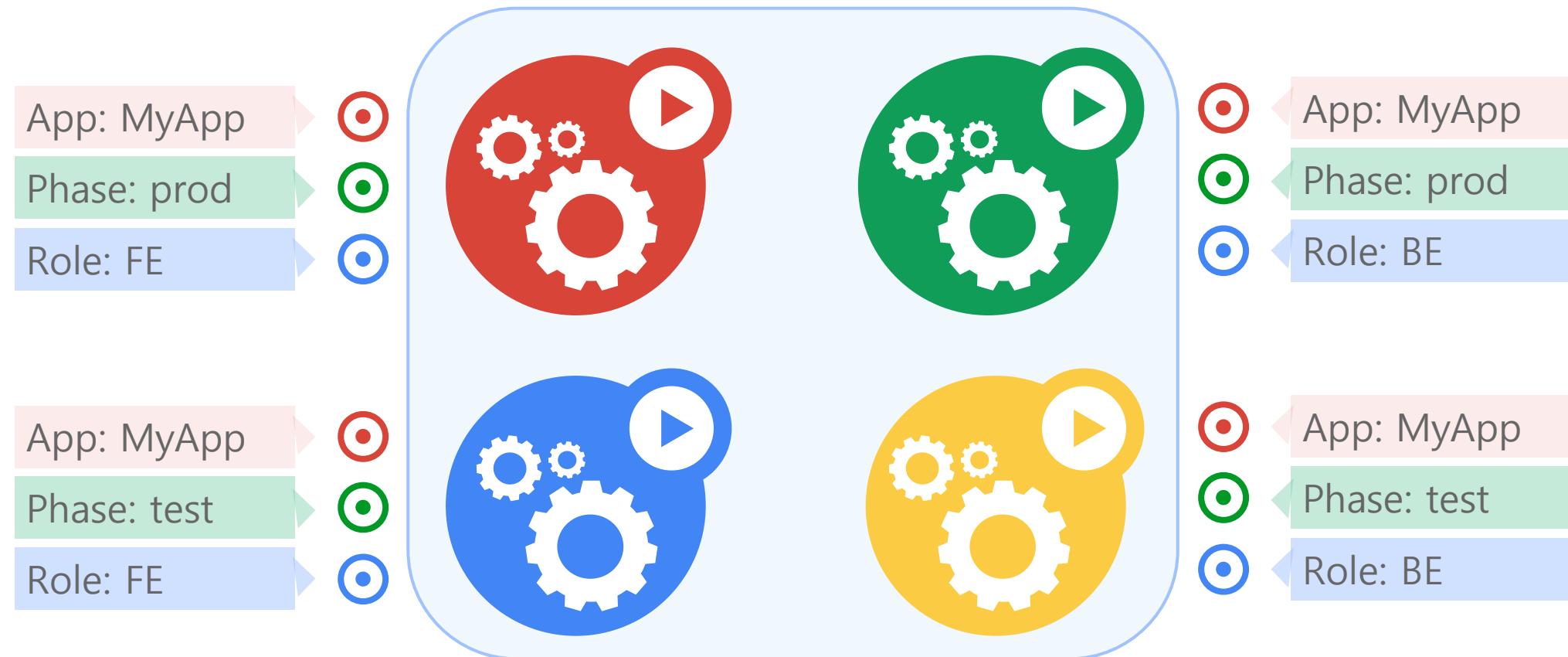
# Labels



# Selectors

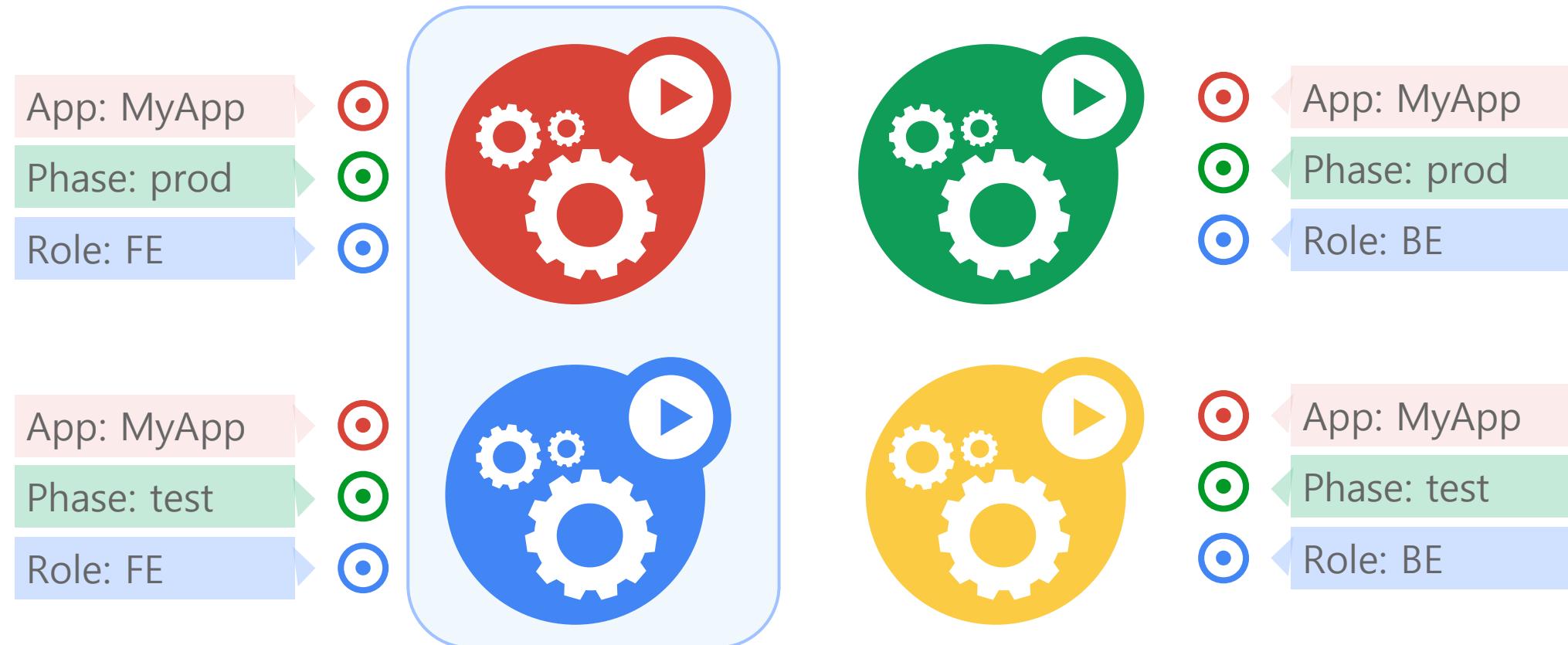


# Selectors



App = MyApp

# Selectors

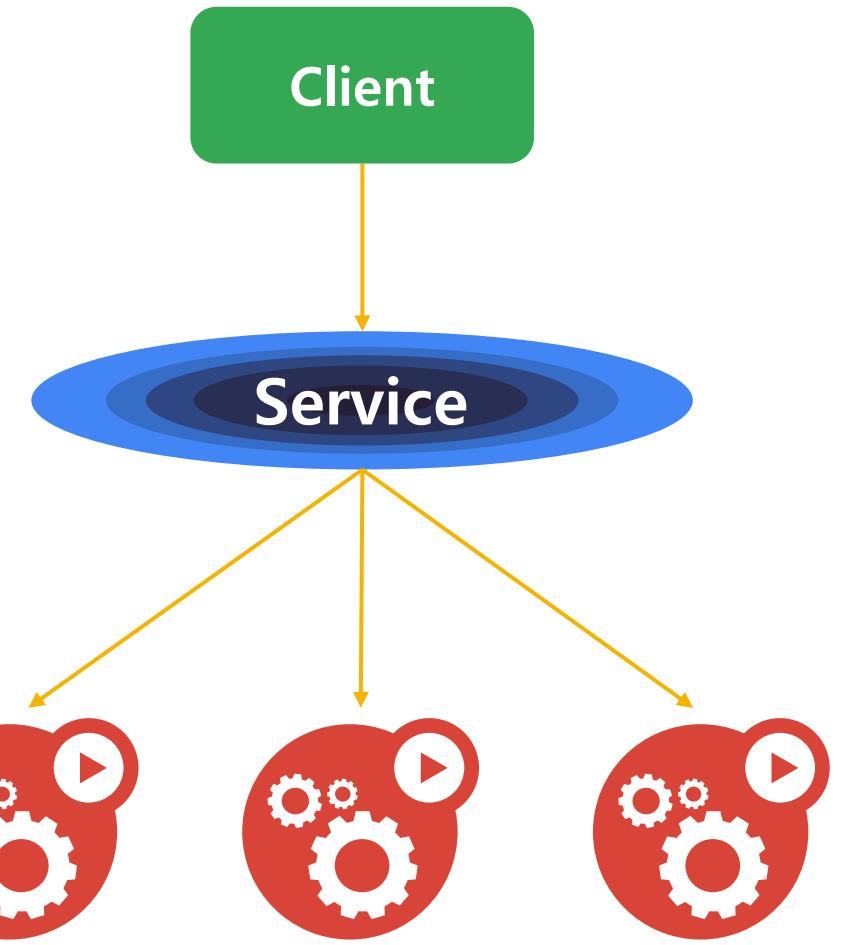


App = MyApp, Role = FE

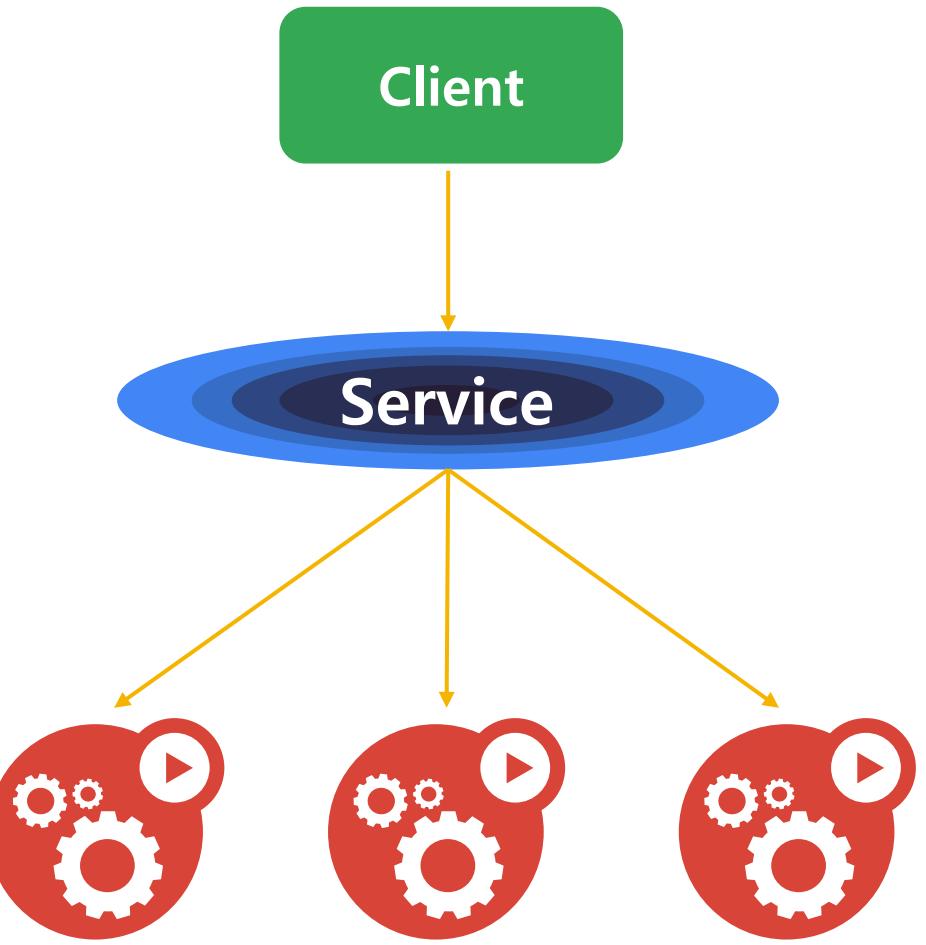
# **Networking Resources**

## **- Services**

# Services



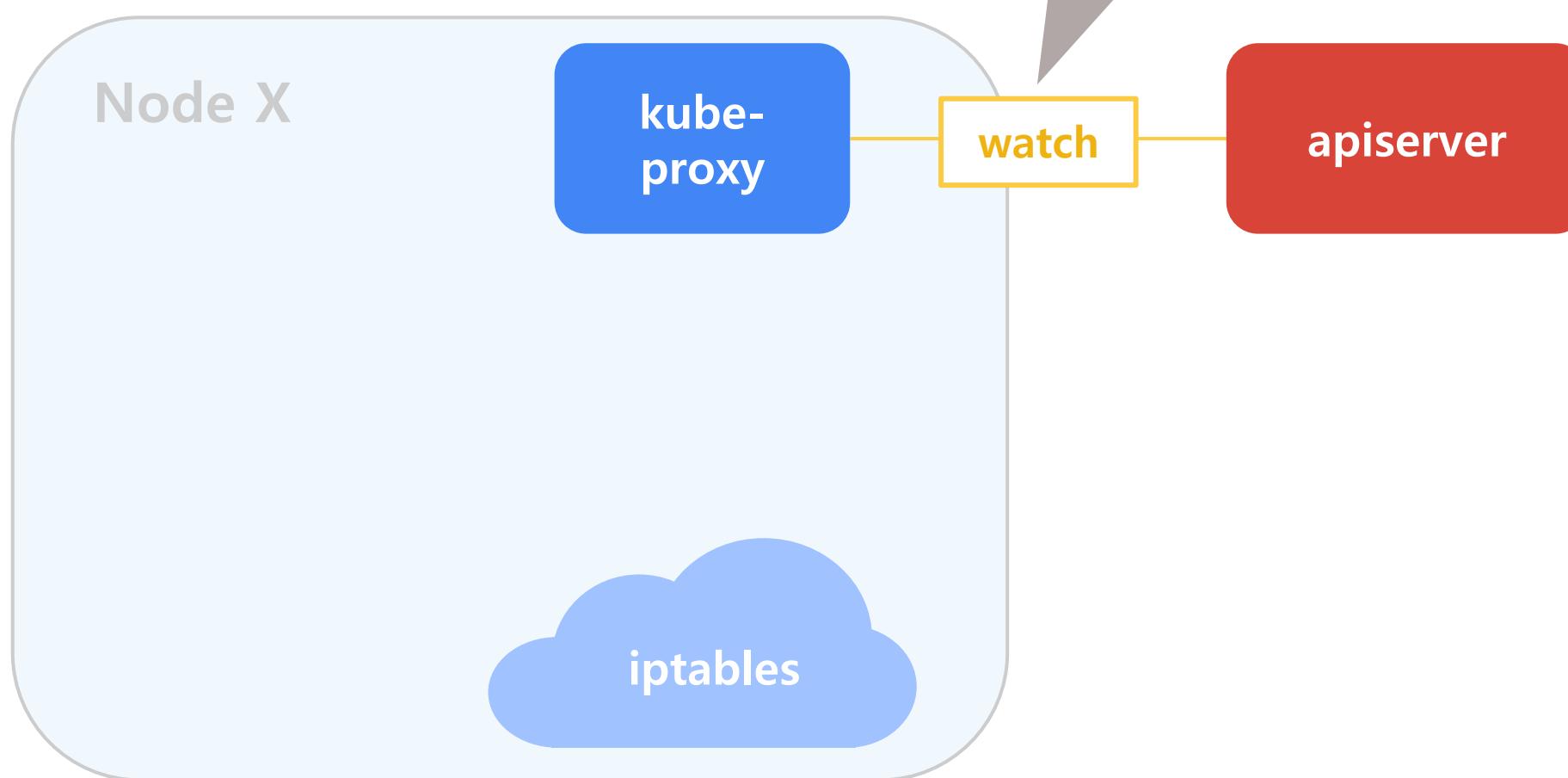
# Service types



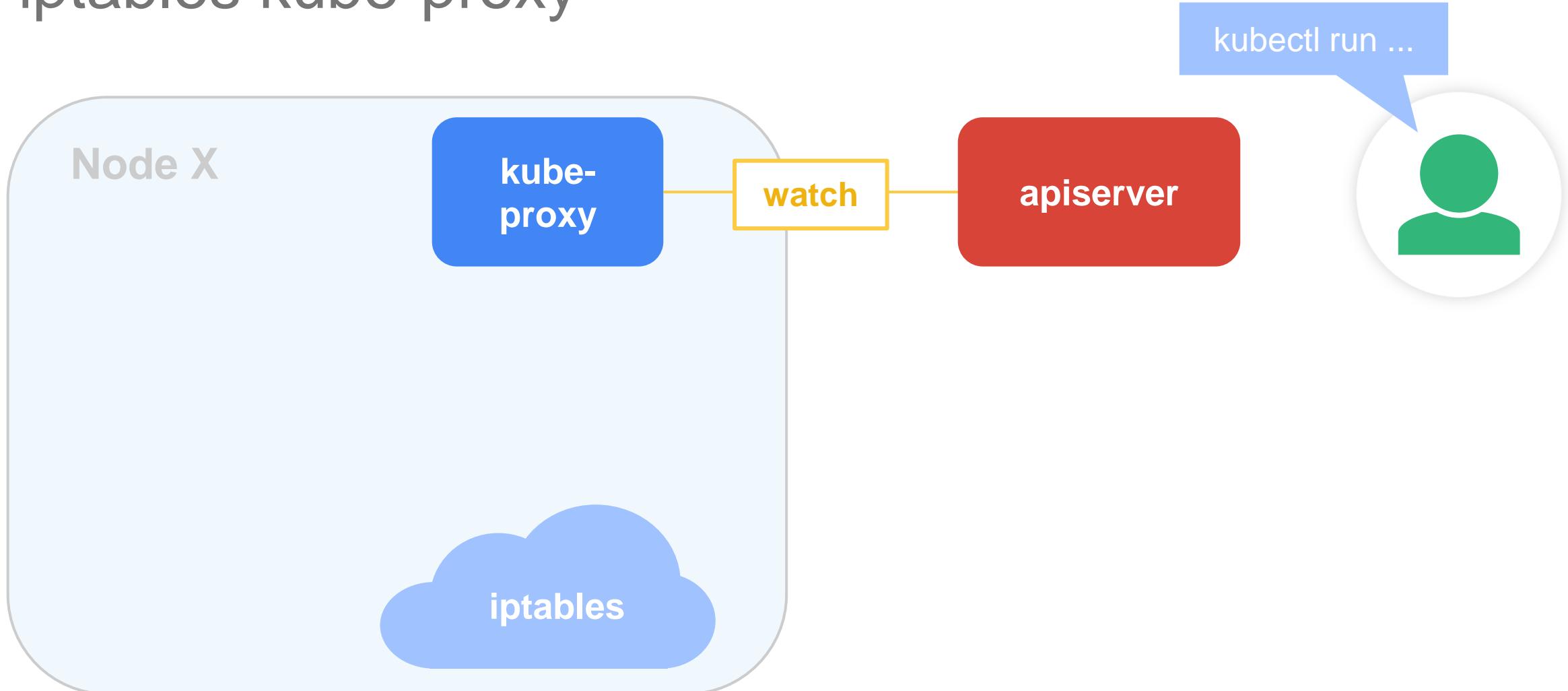
# iptables kube-proxy



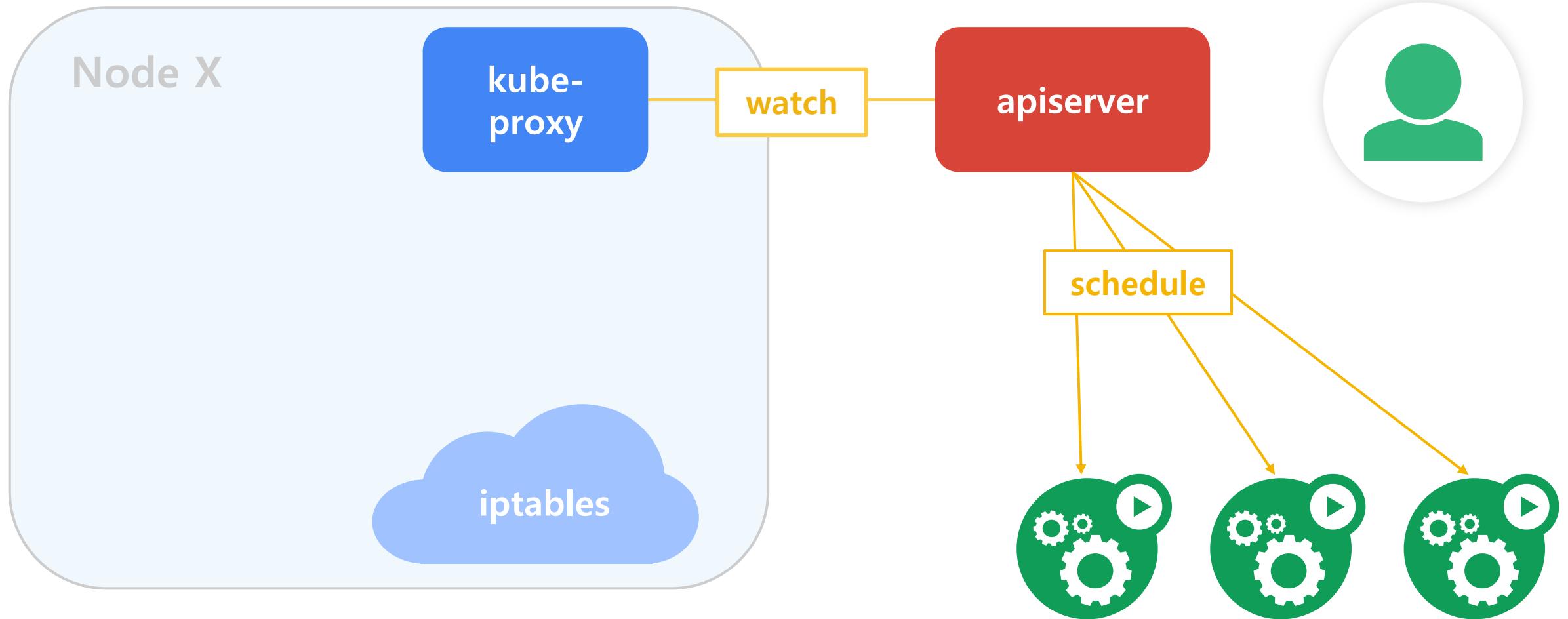
# iptables kube-proxy



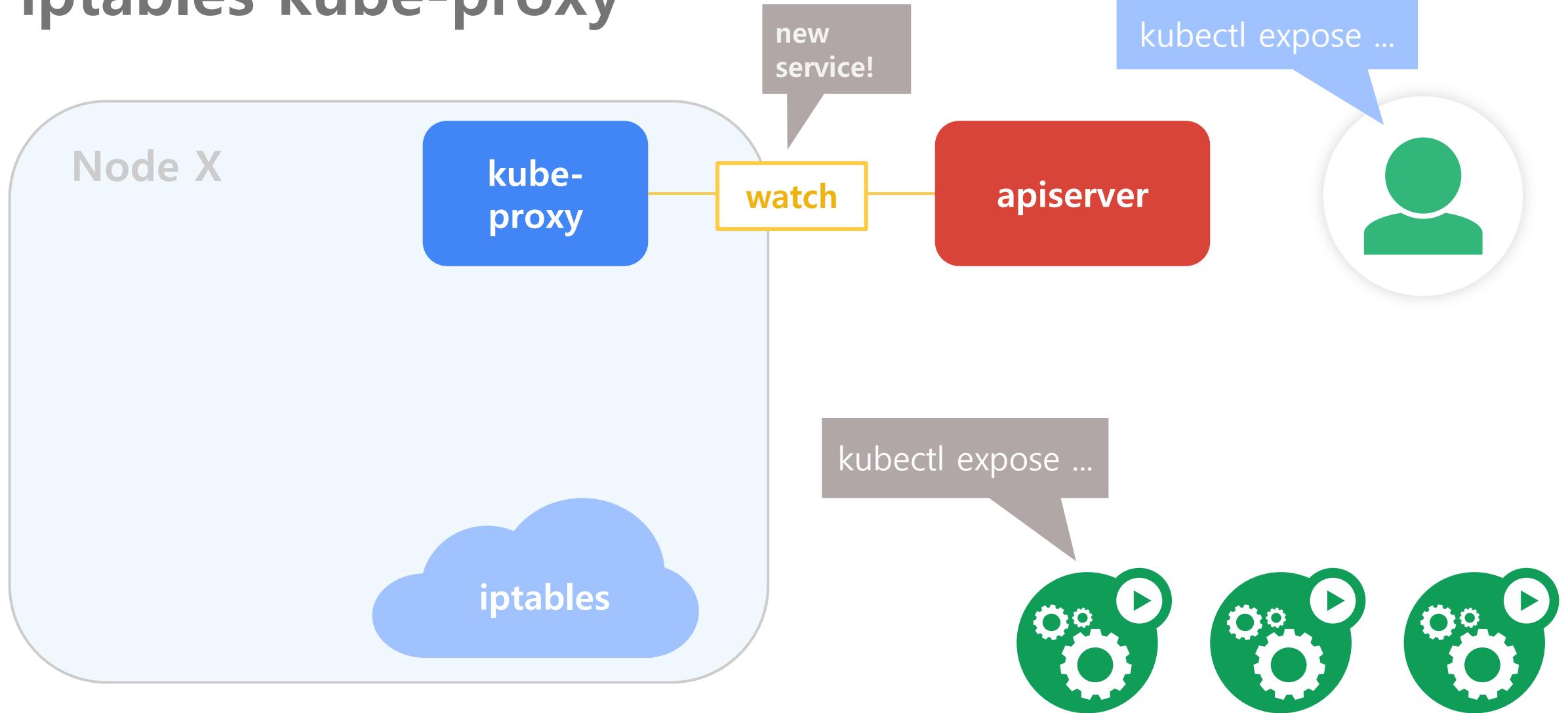
# iptables kube-proxy



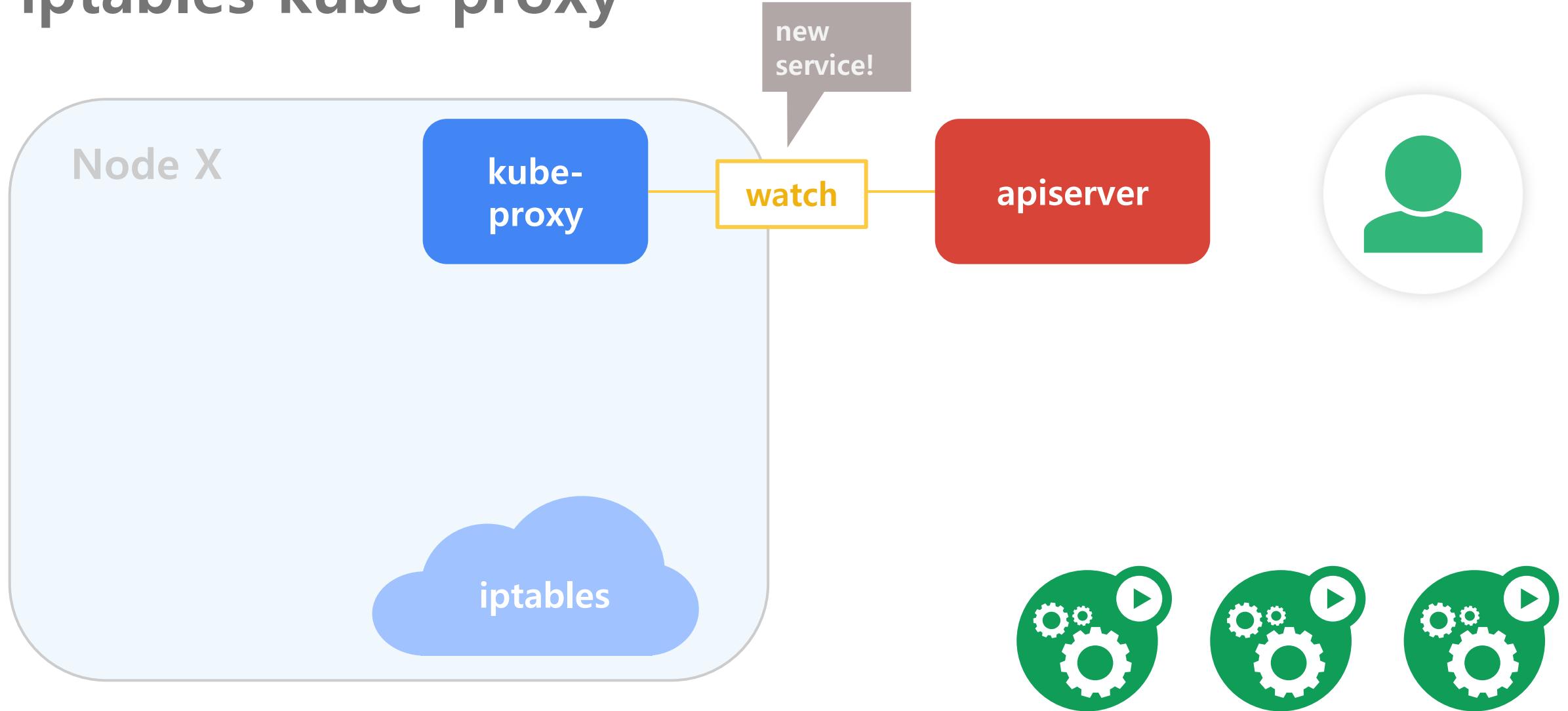
# iptables kube-proxy



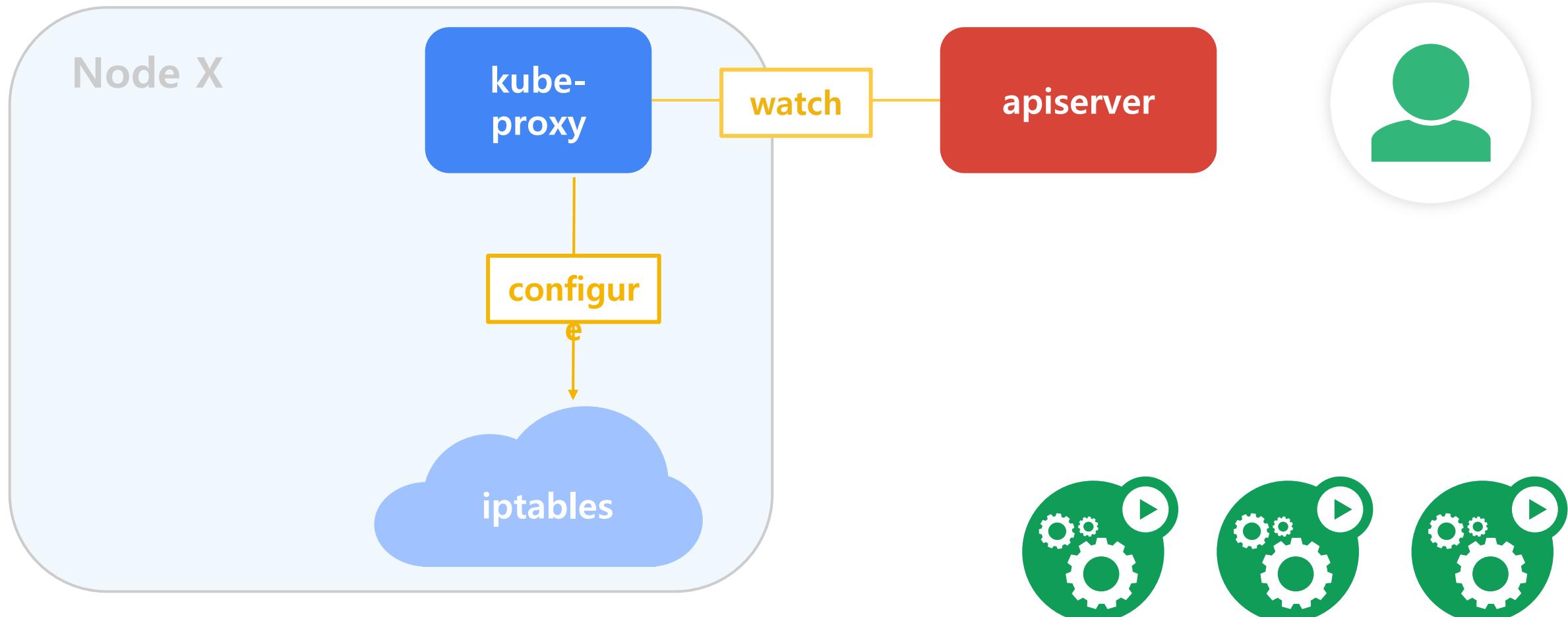
# iptables kube-proxy



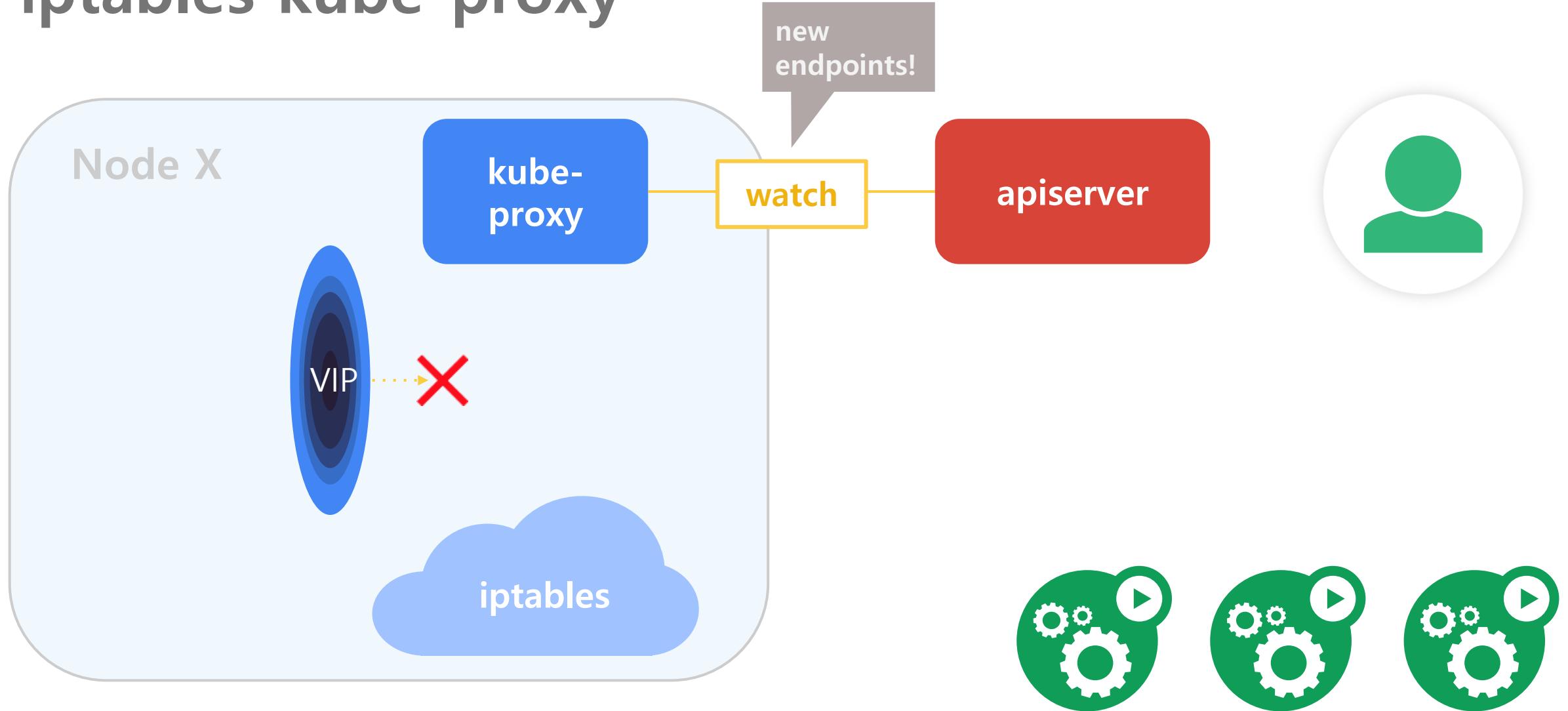
# iptables kube-proxy



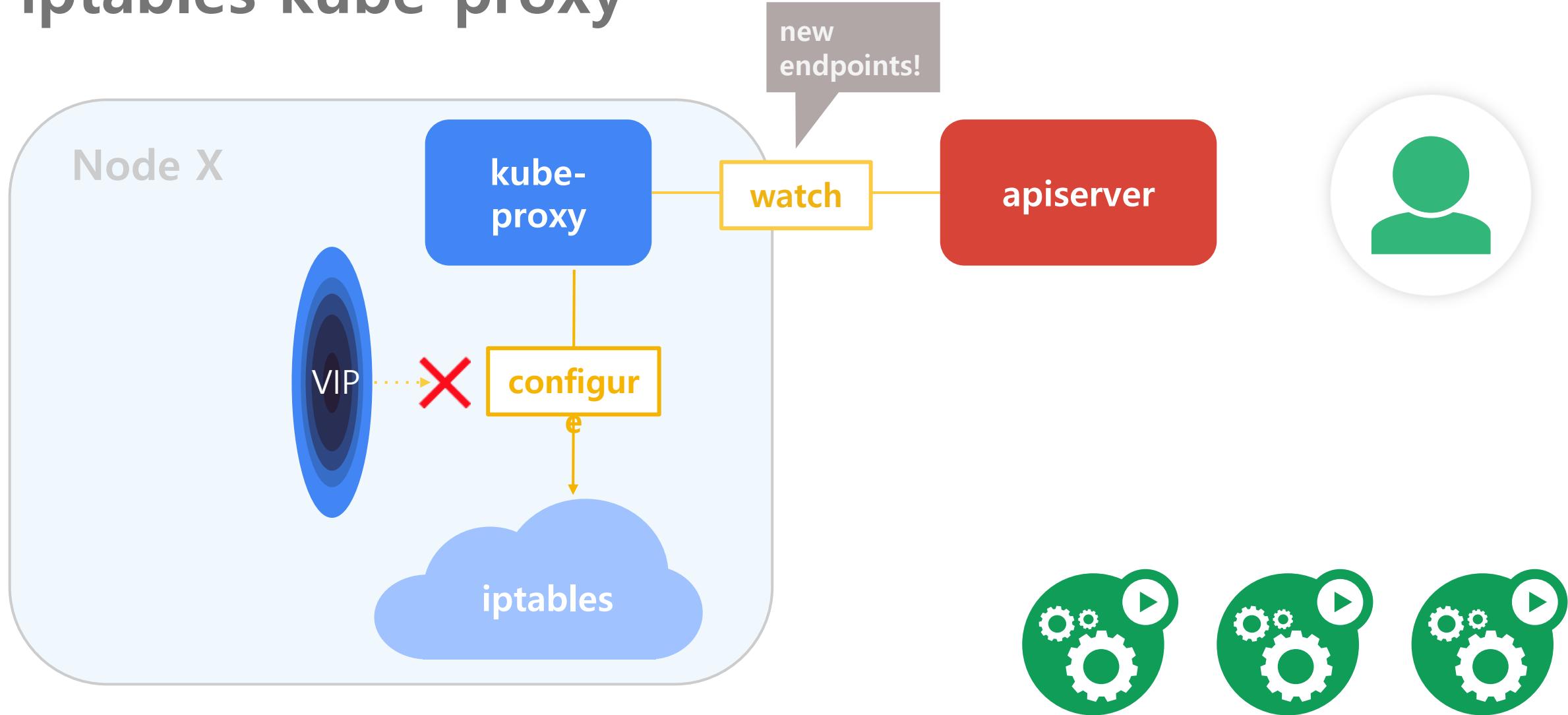
# iptables kube-proxy



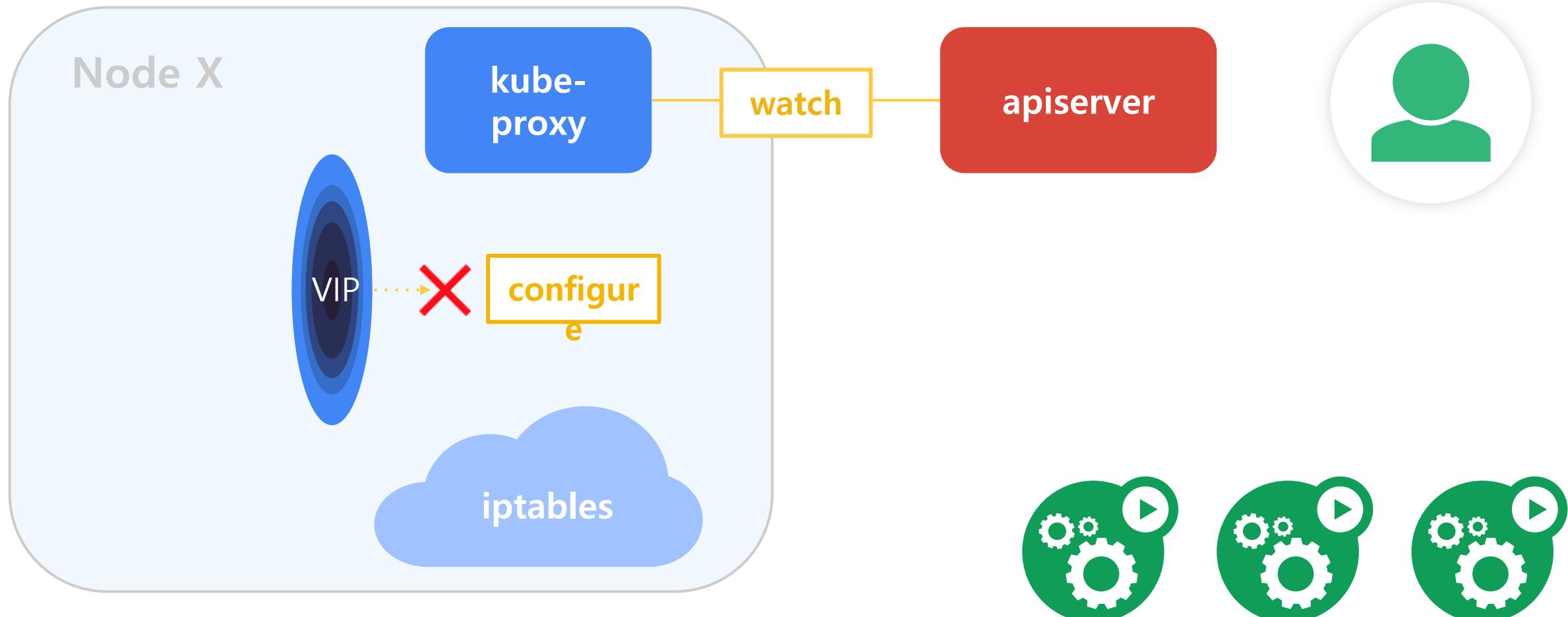
# iptables kube-proxy



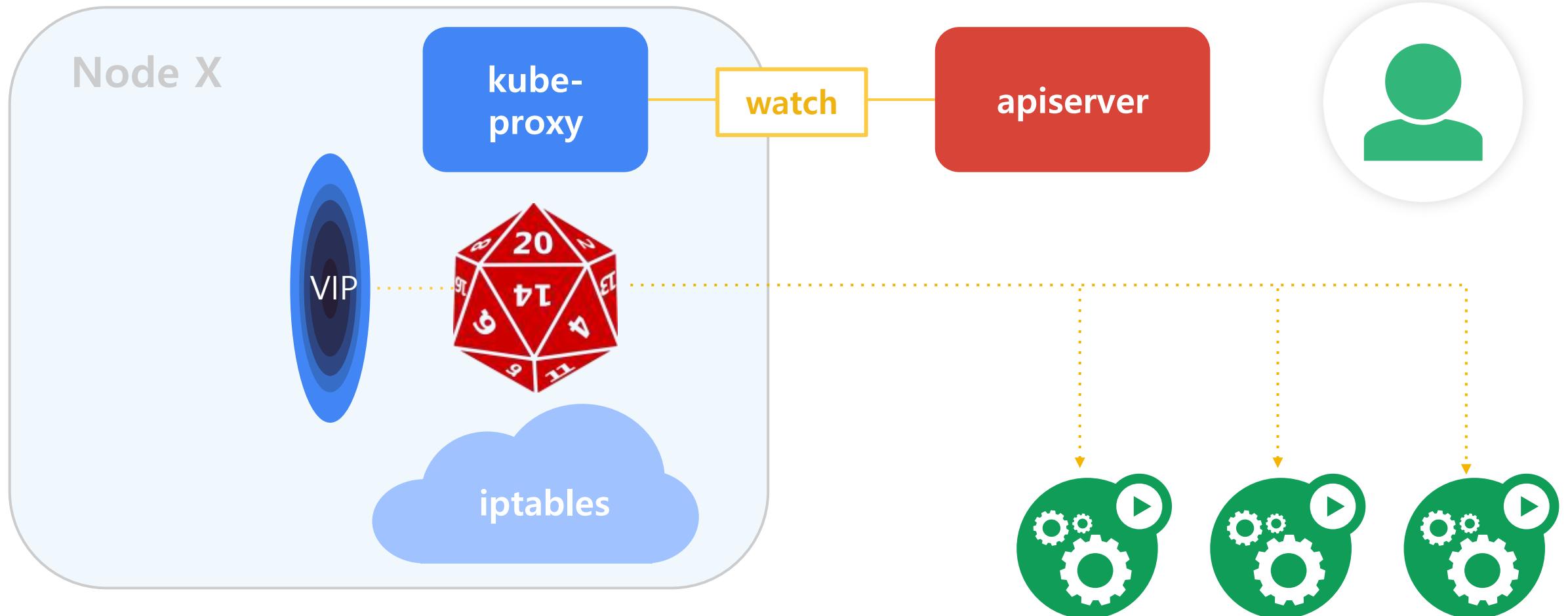
# iptables kube-proxy



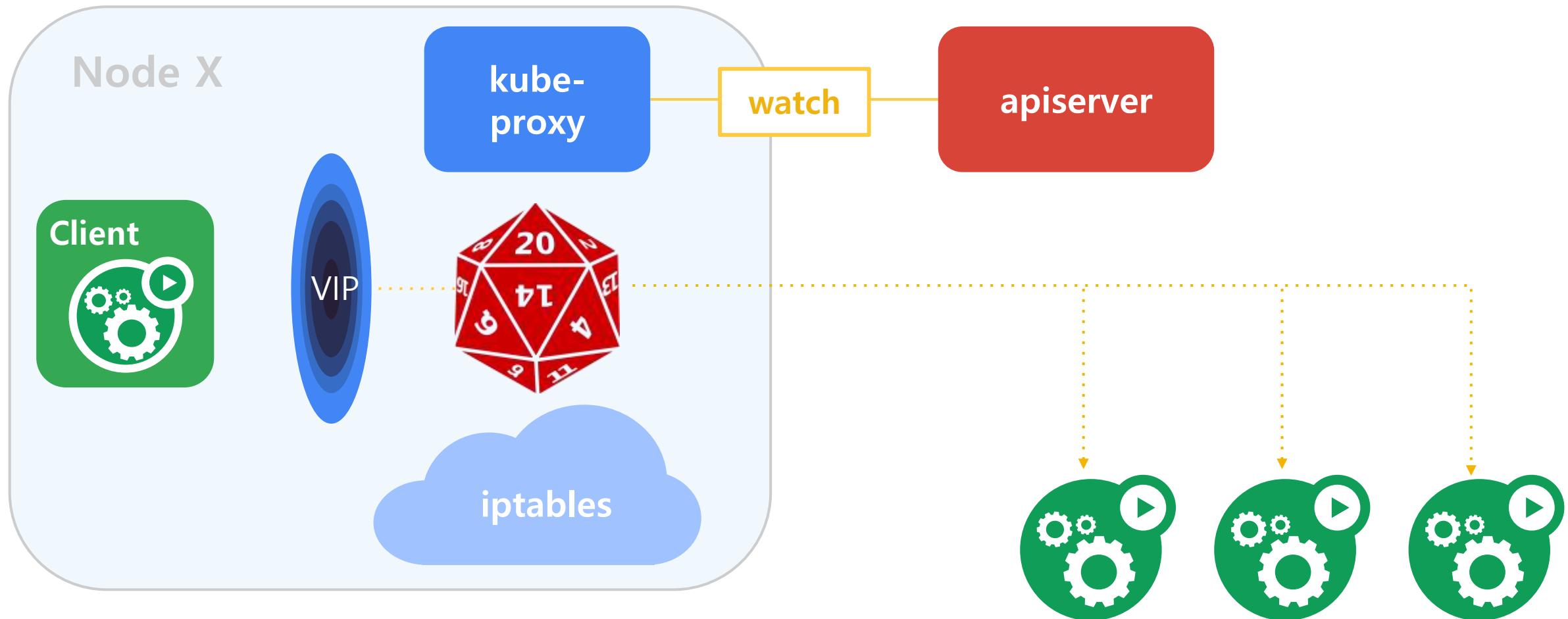
# iptables kube-proxy



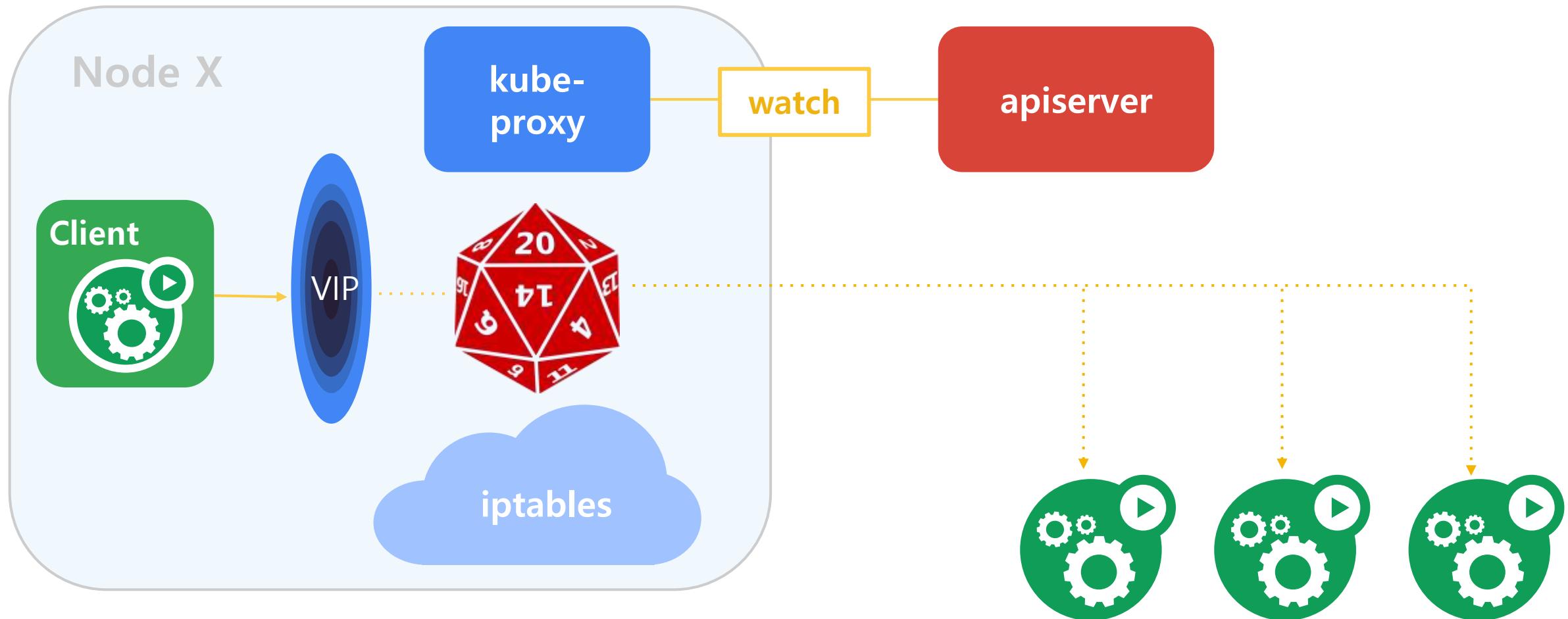
# iptables kube-proxy



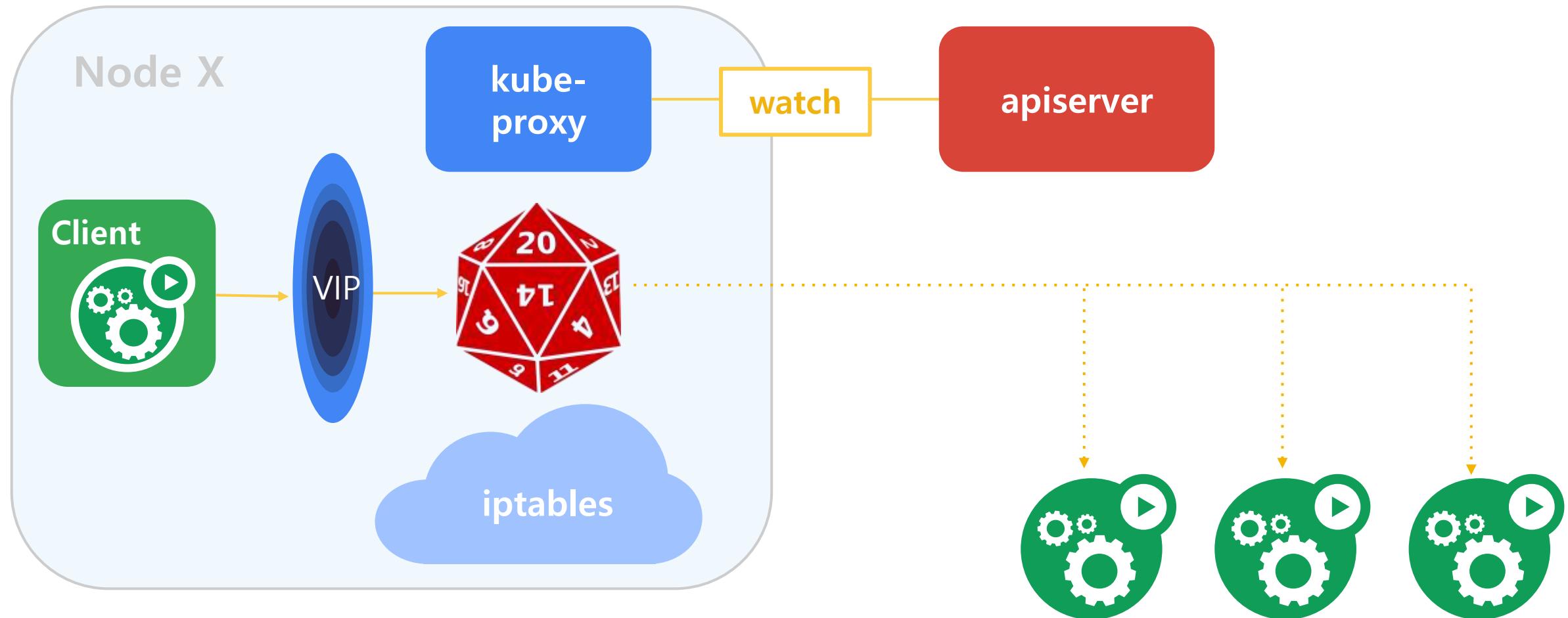
# iptables kube-proxy



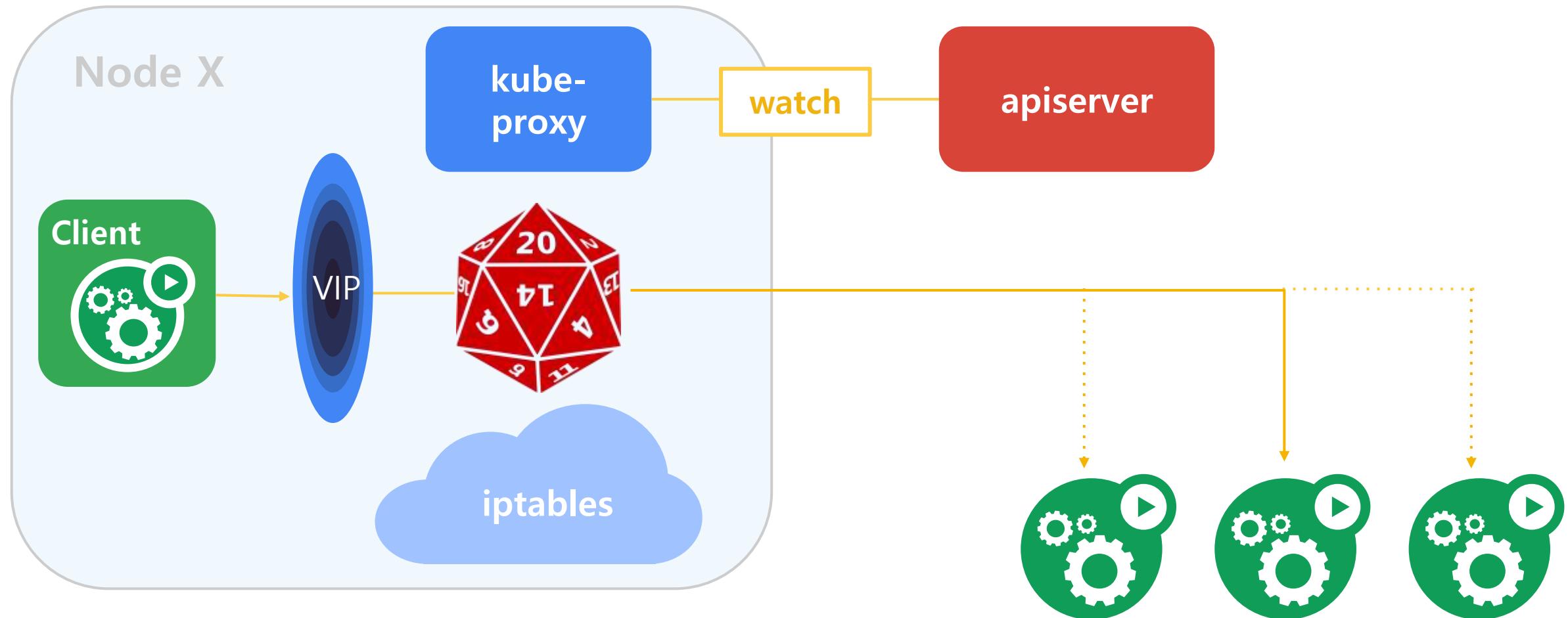
# iptables kube-proxy



# iptables kube-proxy



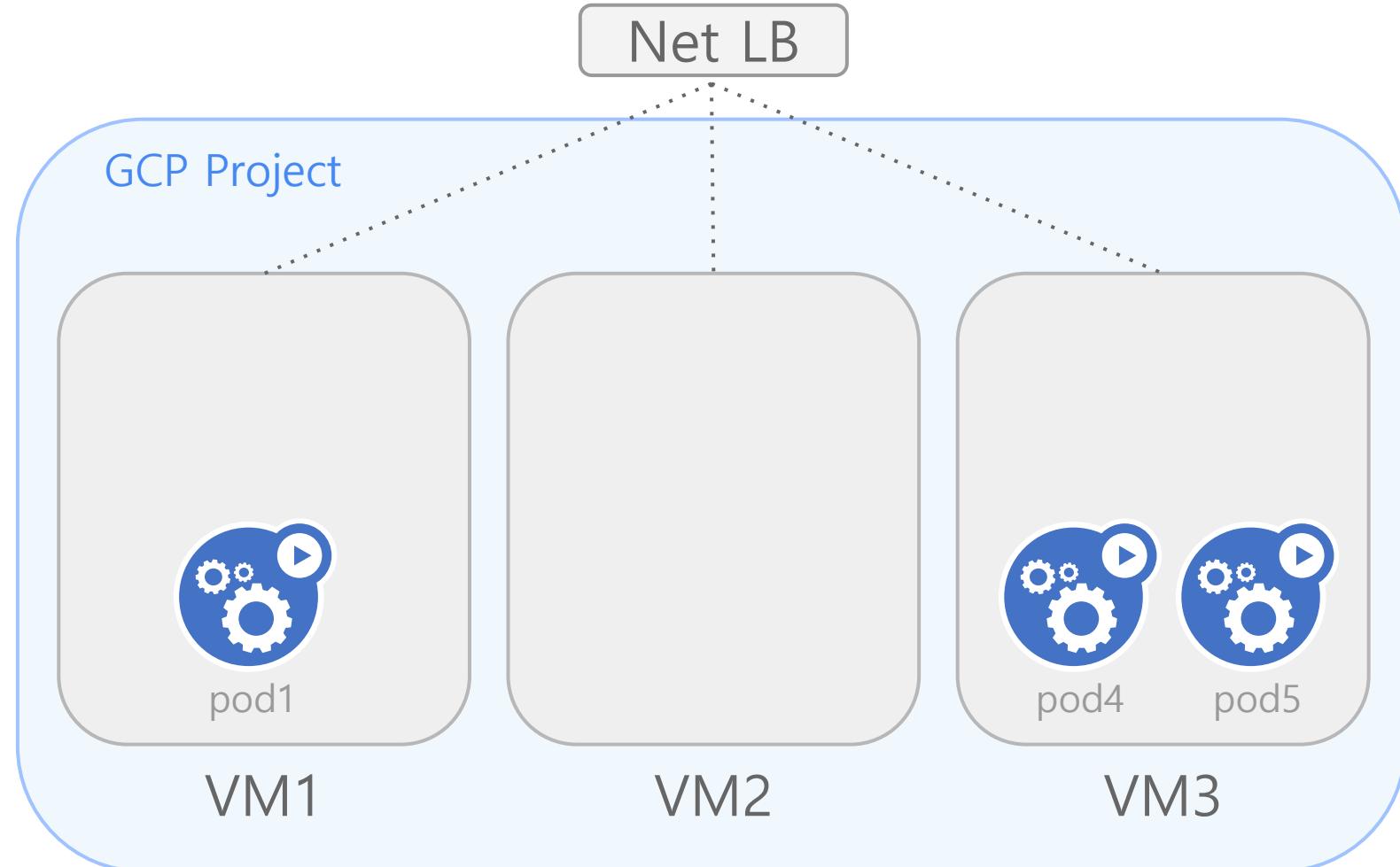
# iptables kube-proxy



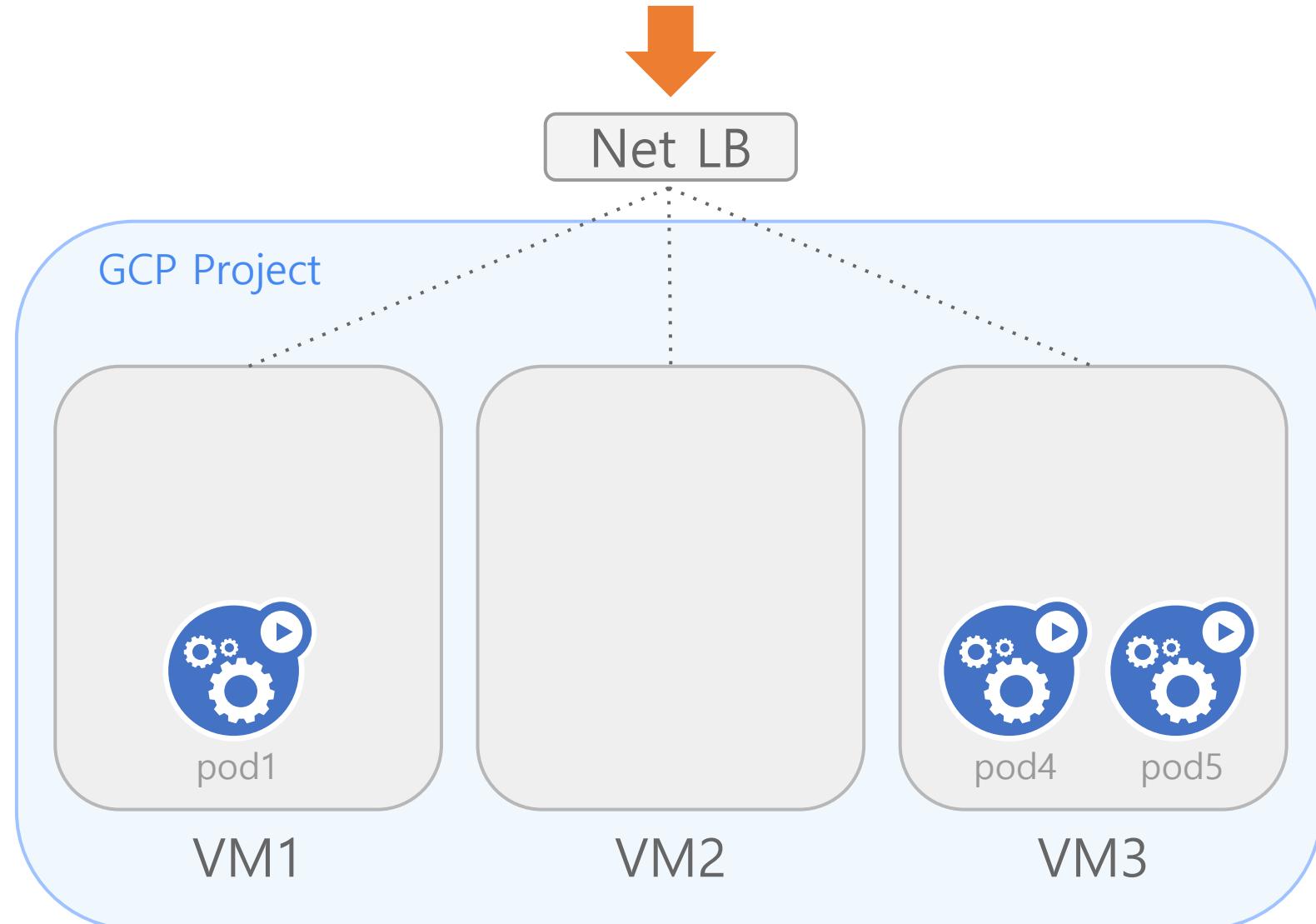
# Service type: NodePort



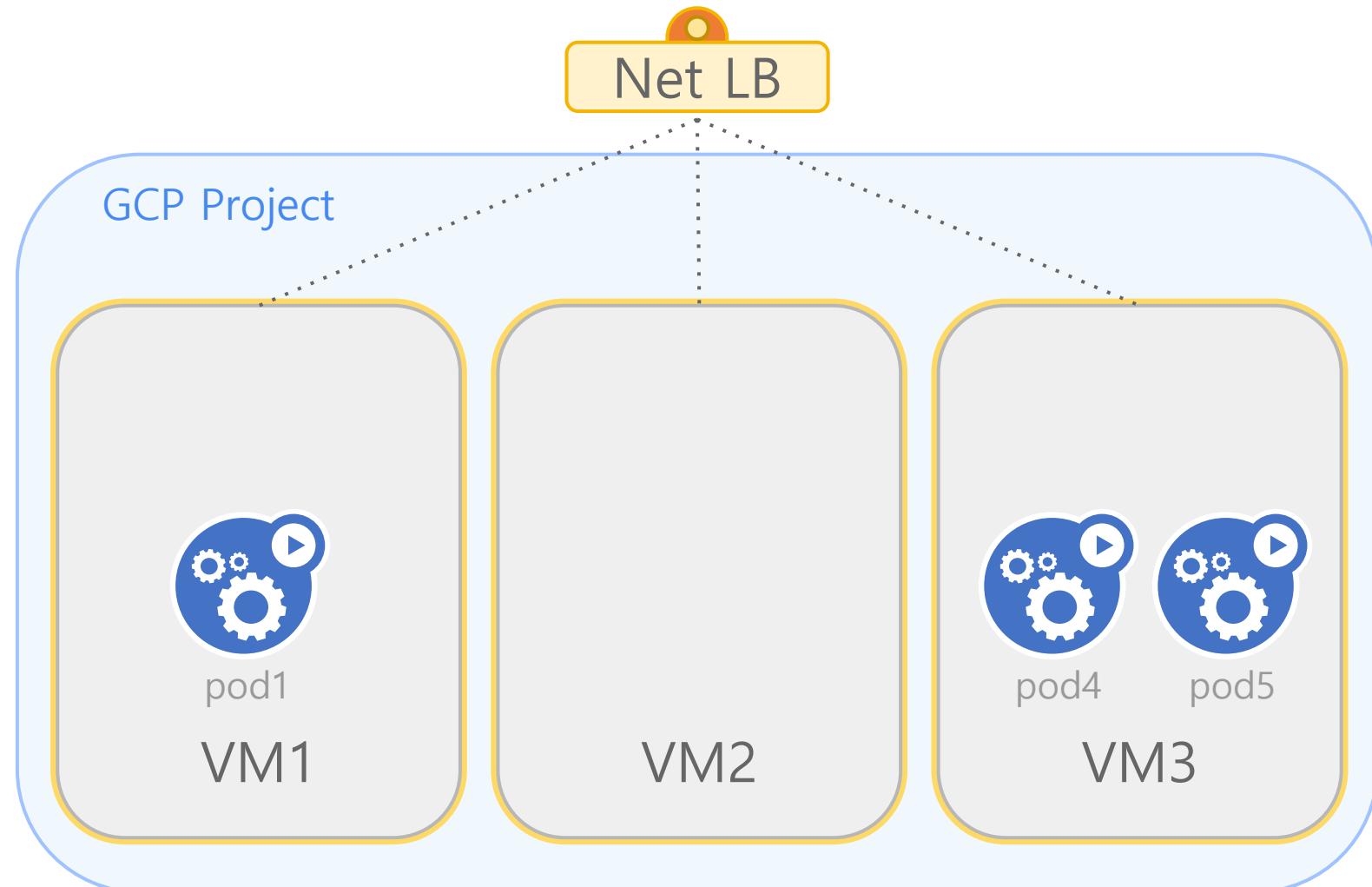
# Service type: LoadBalancer



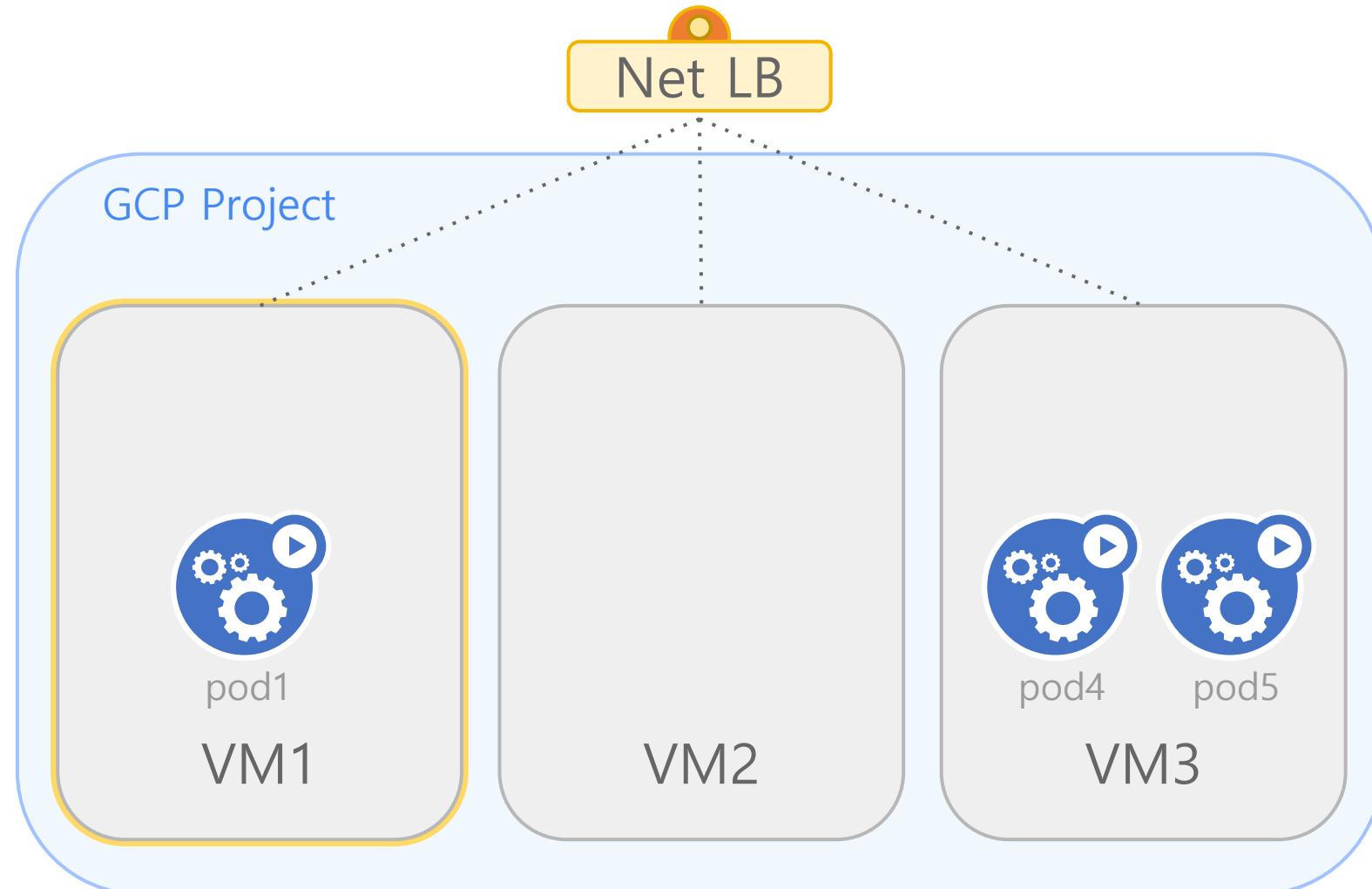
# Service type: LoadBalancer



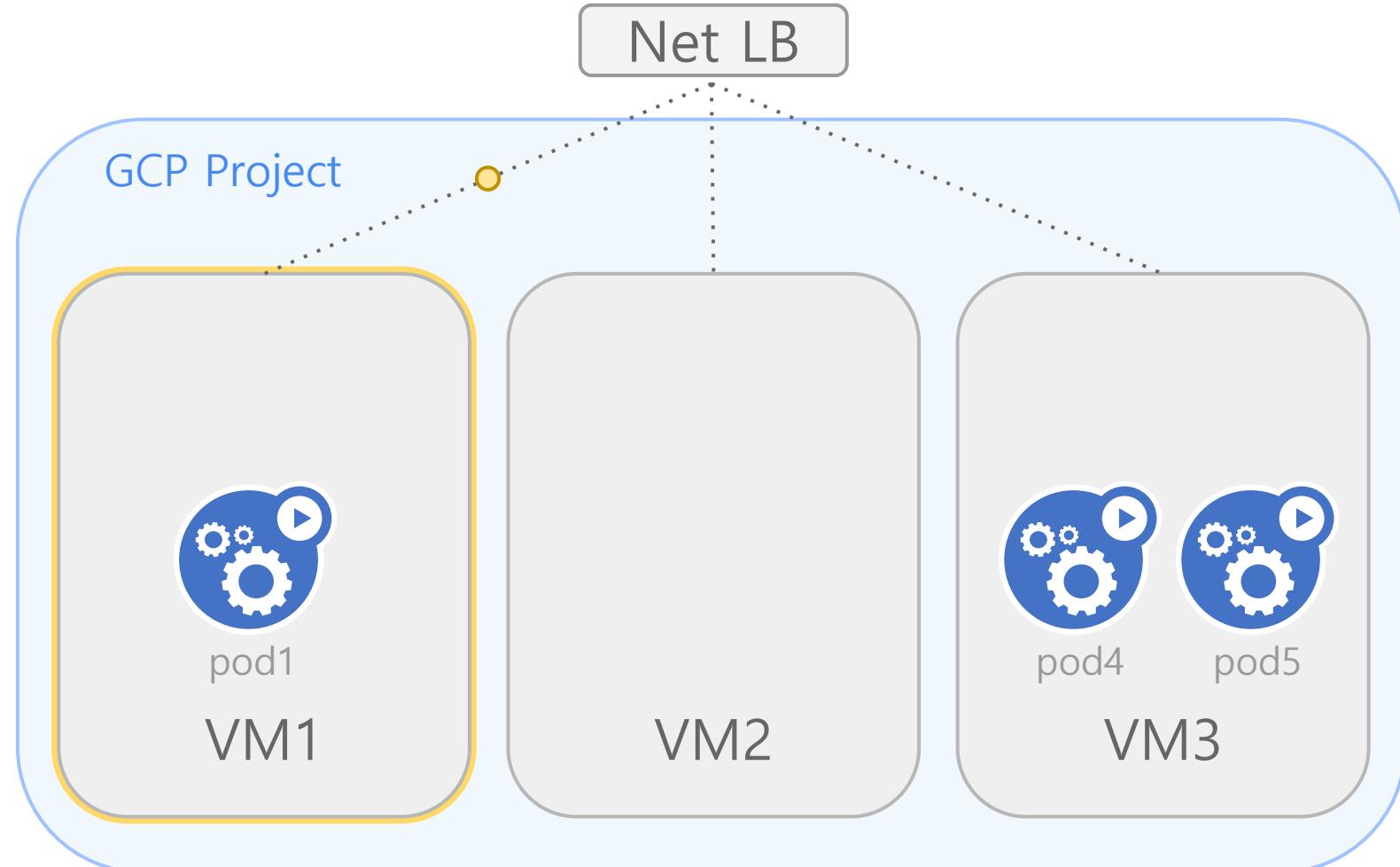
# Service type: LoadBalancer



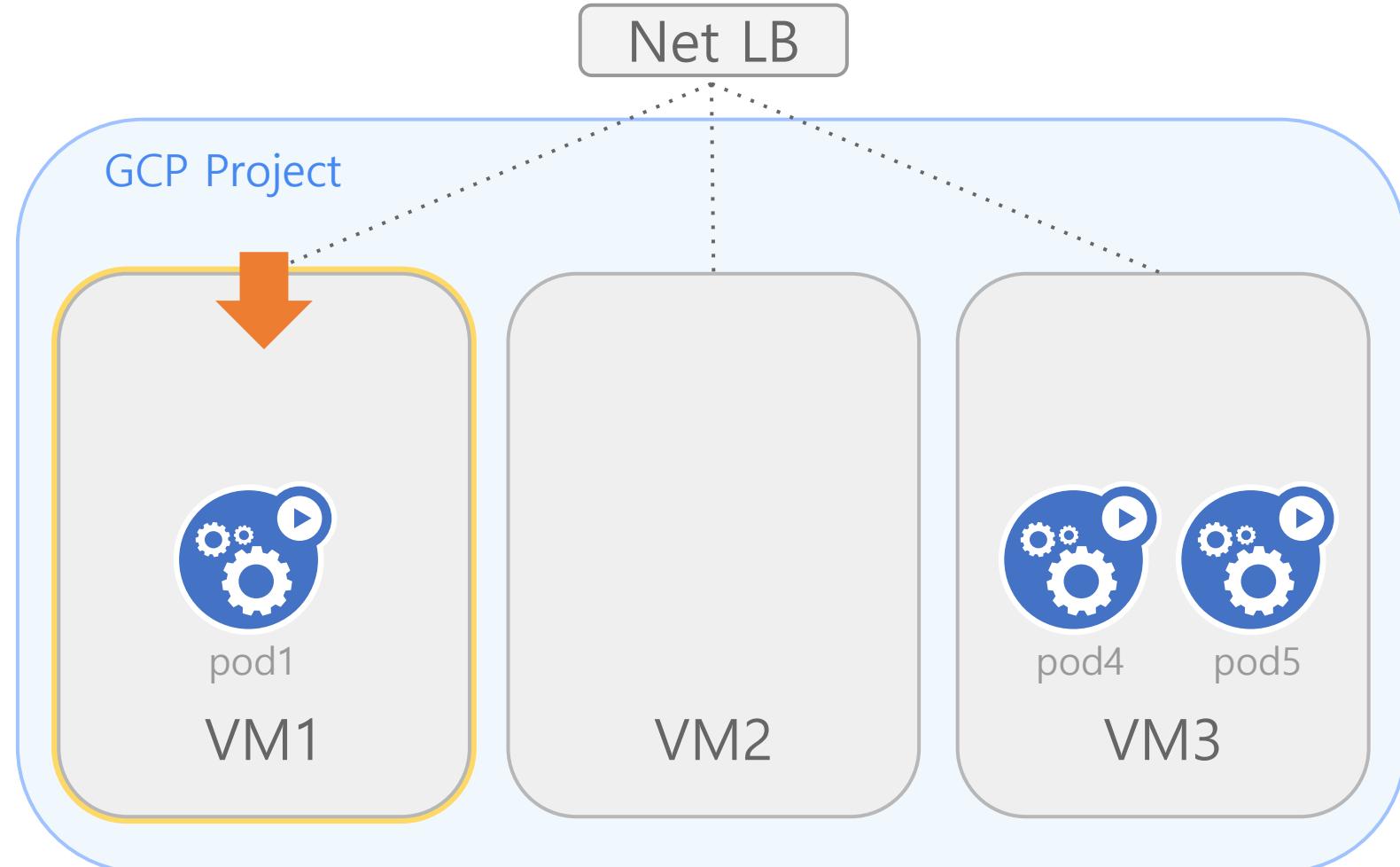
# Service type: LoadBalancer



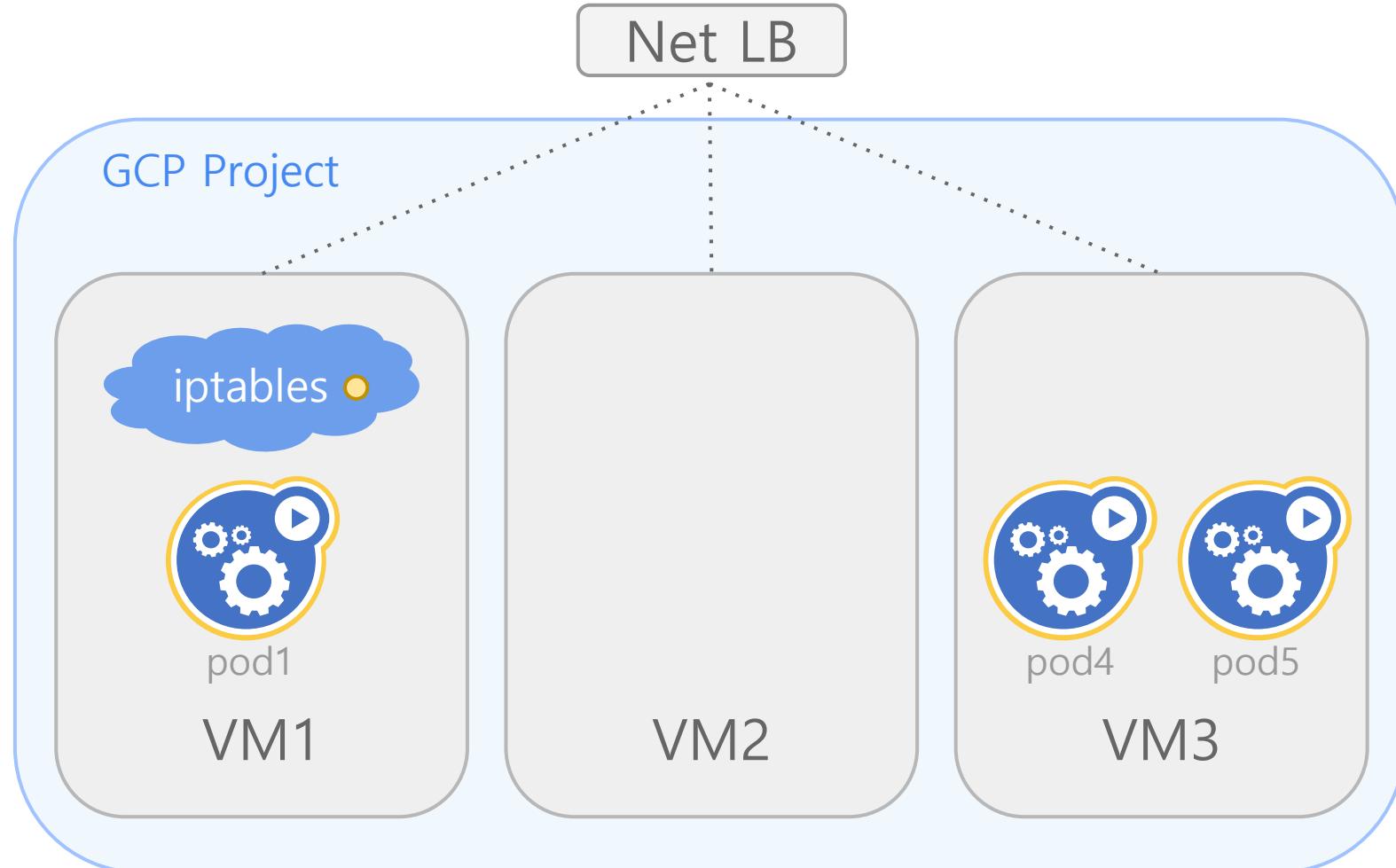
# Service type: LoadBalancer



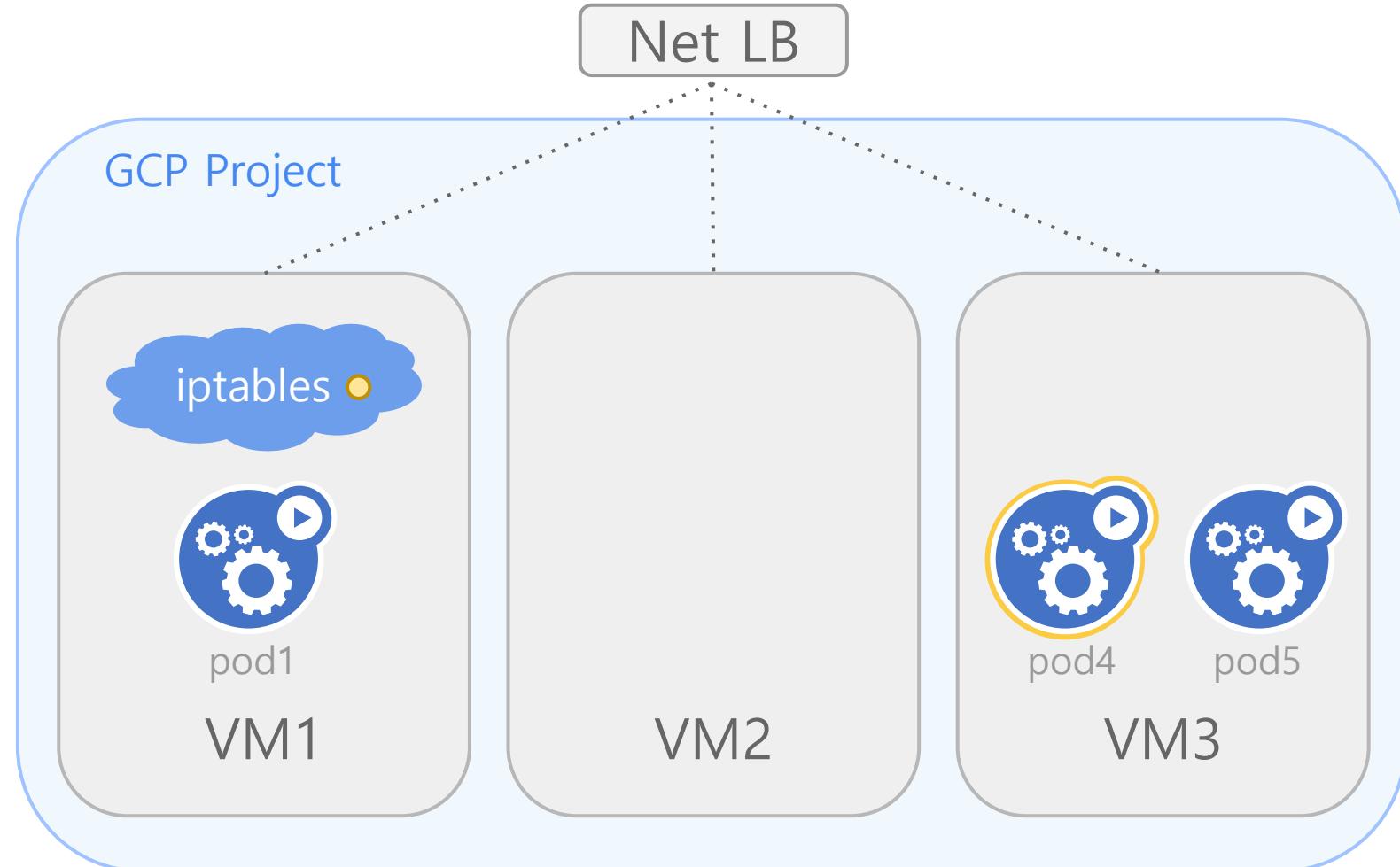
# Service type: LoadBalancer



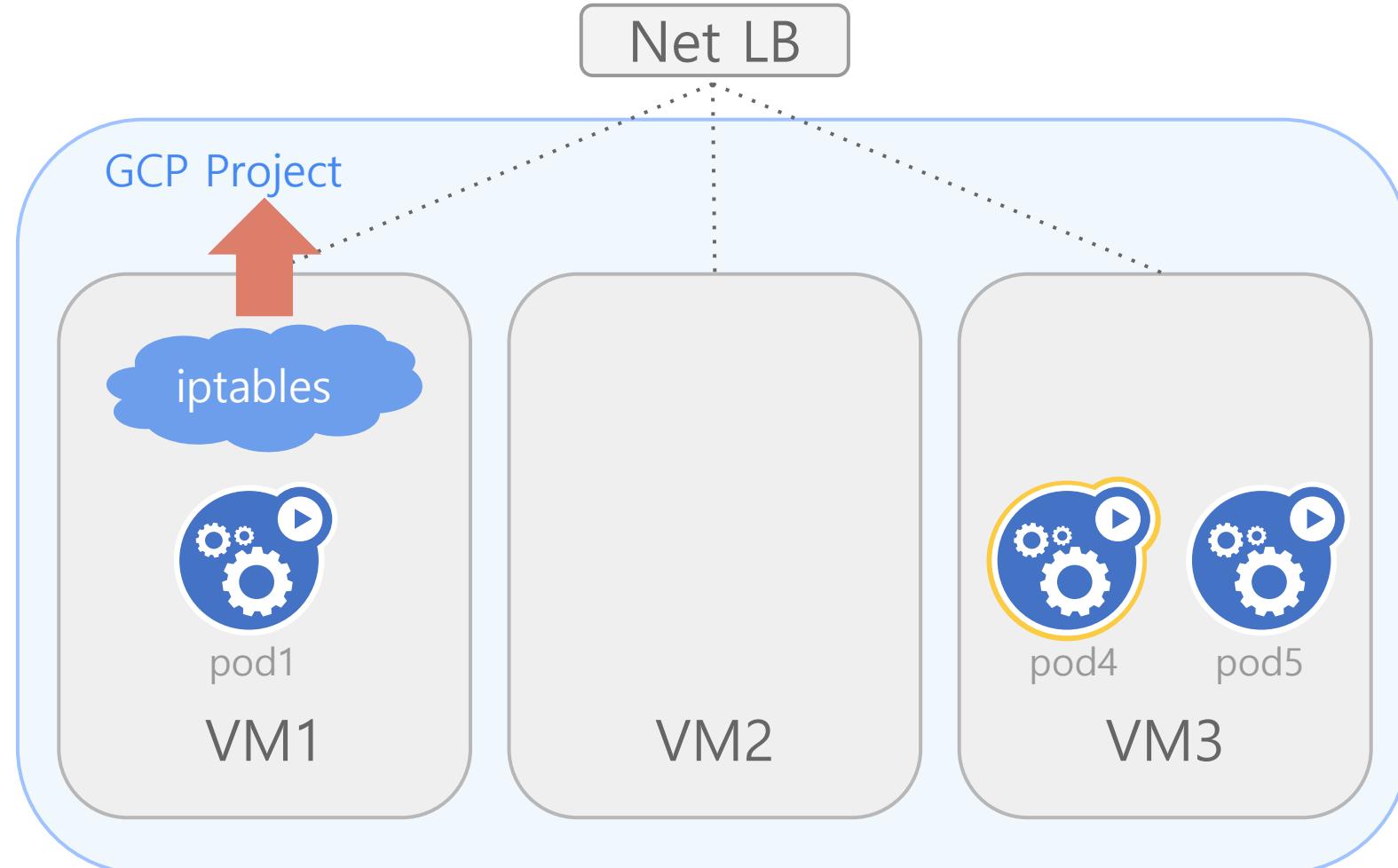
# Service type: LoadBalancer



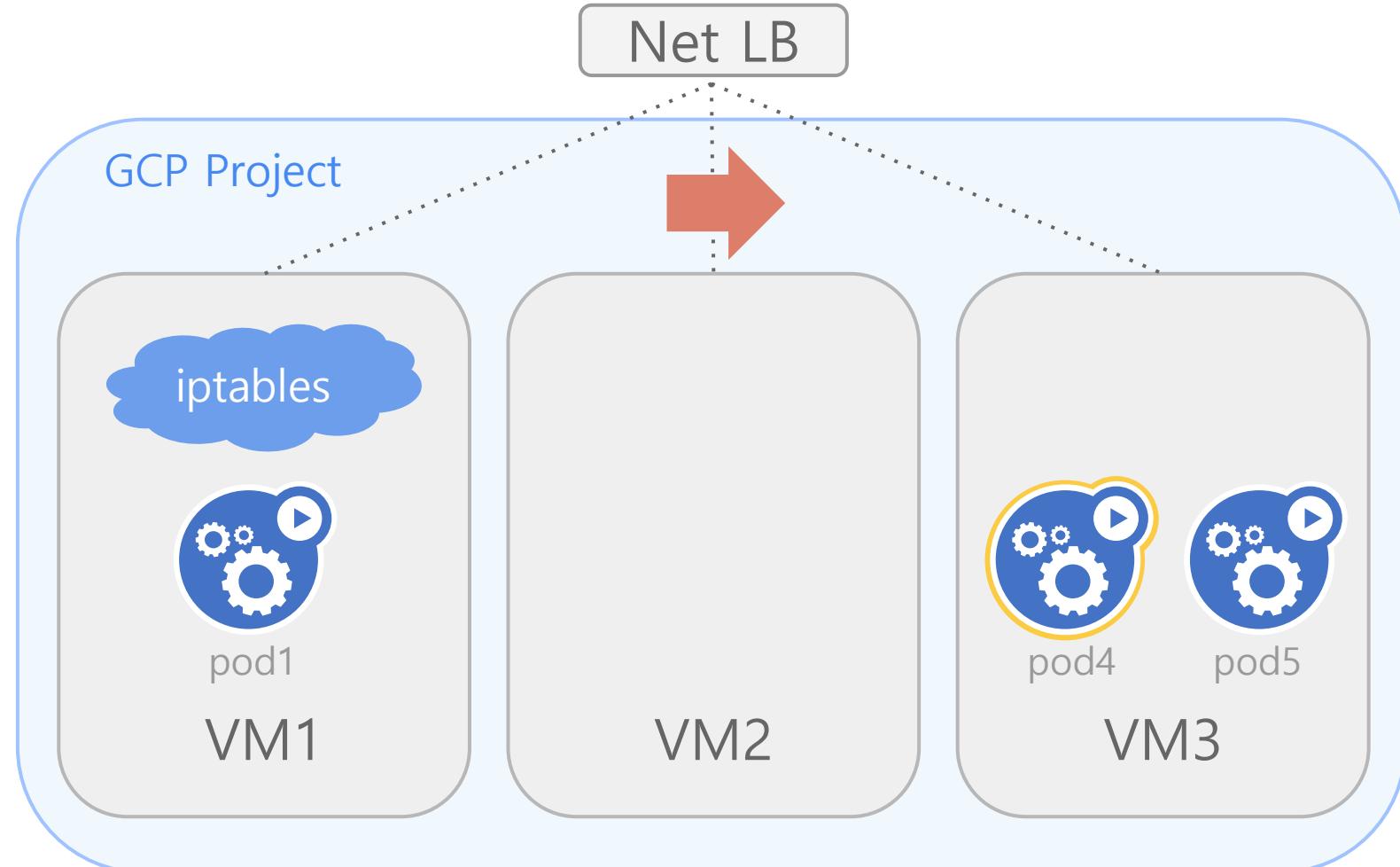
# Service type: LoadBalancer



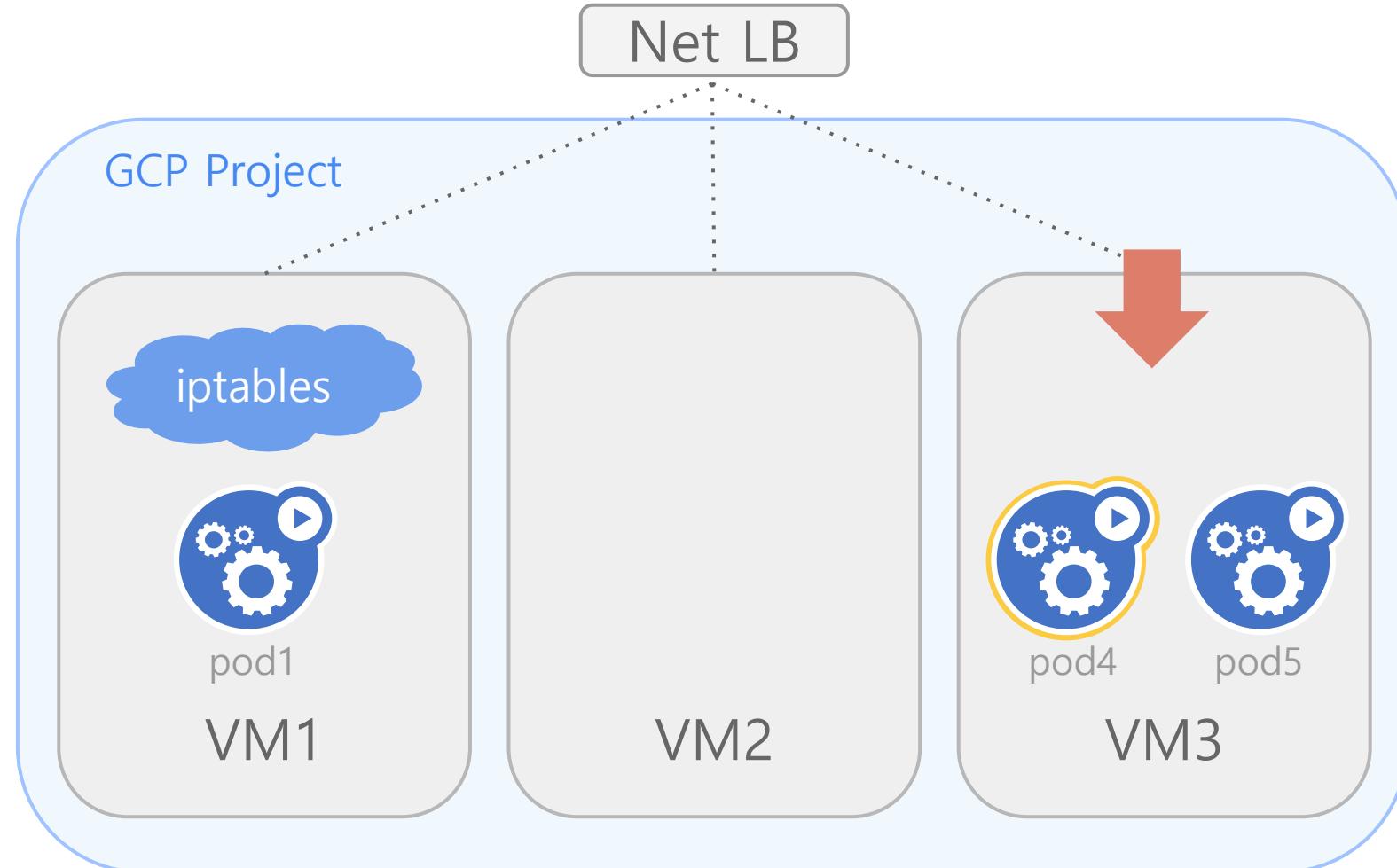
# Service type: LoadBalancer



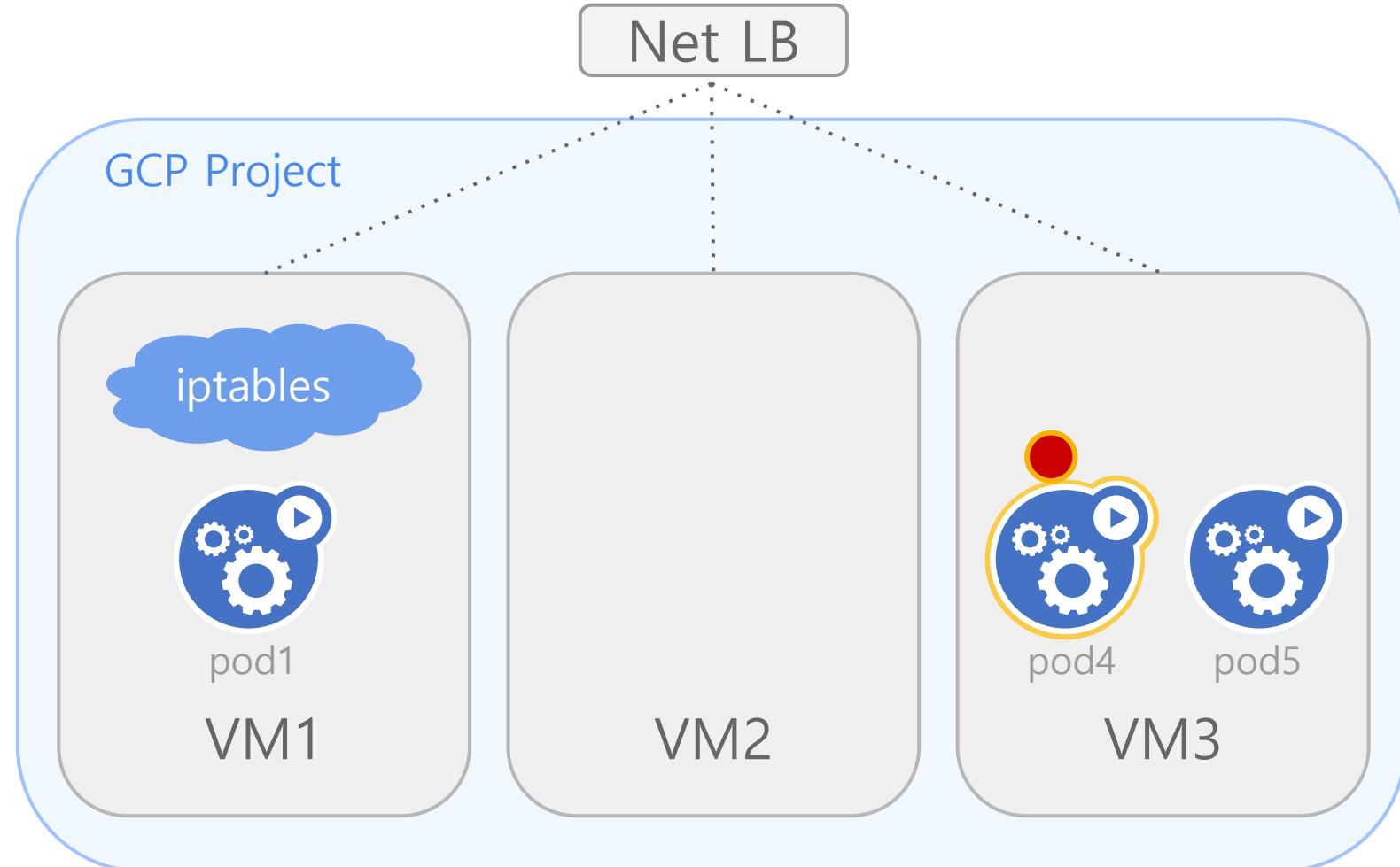
# Service type: LoadBalancer



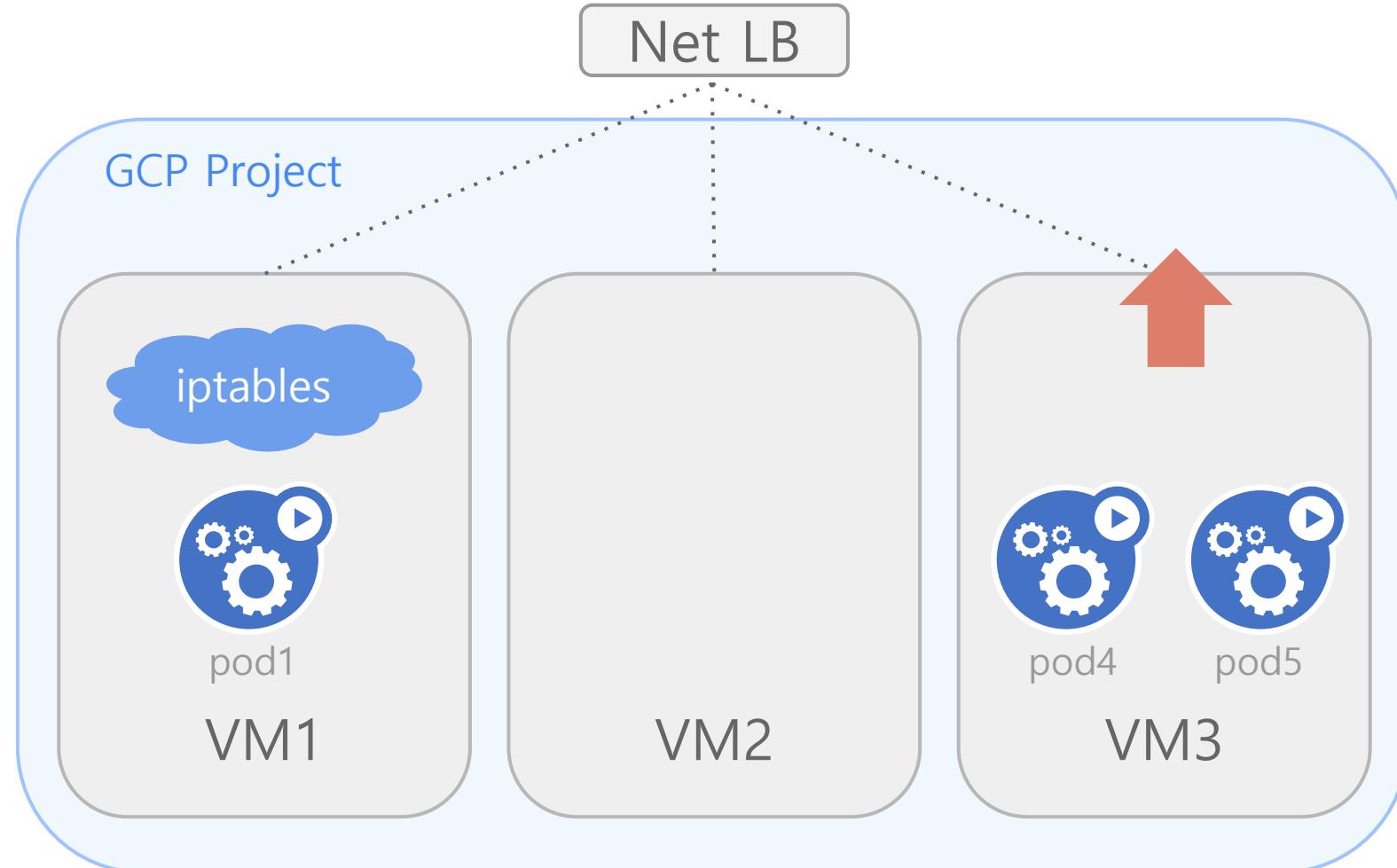
# Service type: LoadBalancer



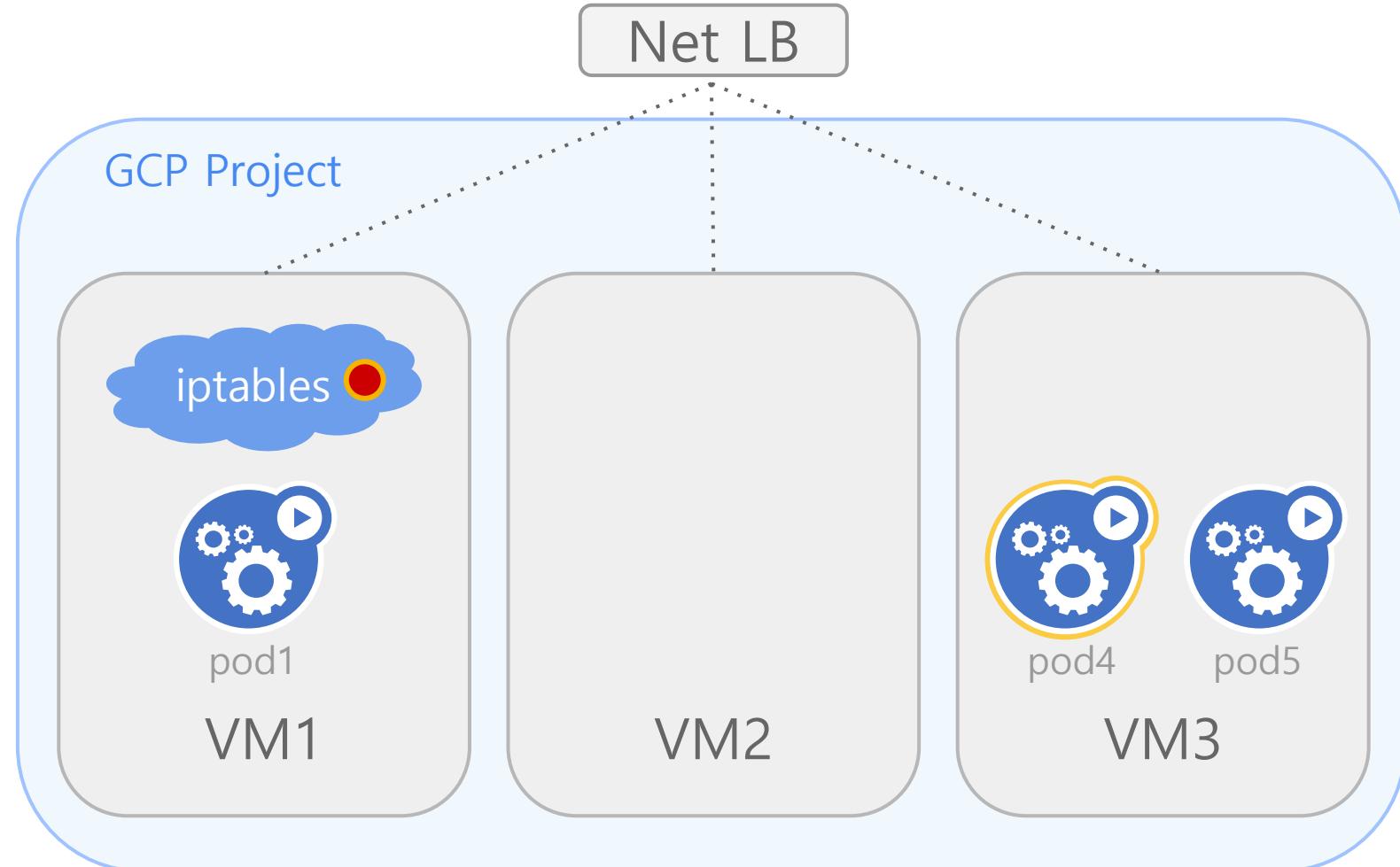
# Service type: LoadBalancer



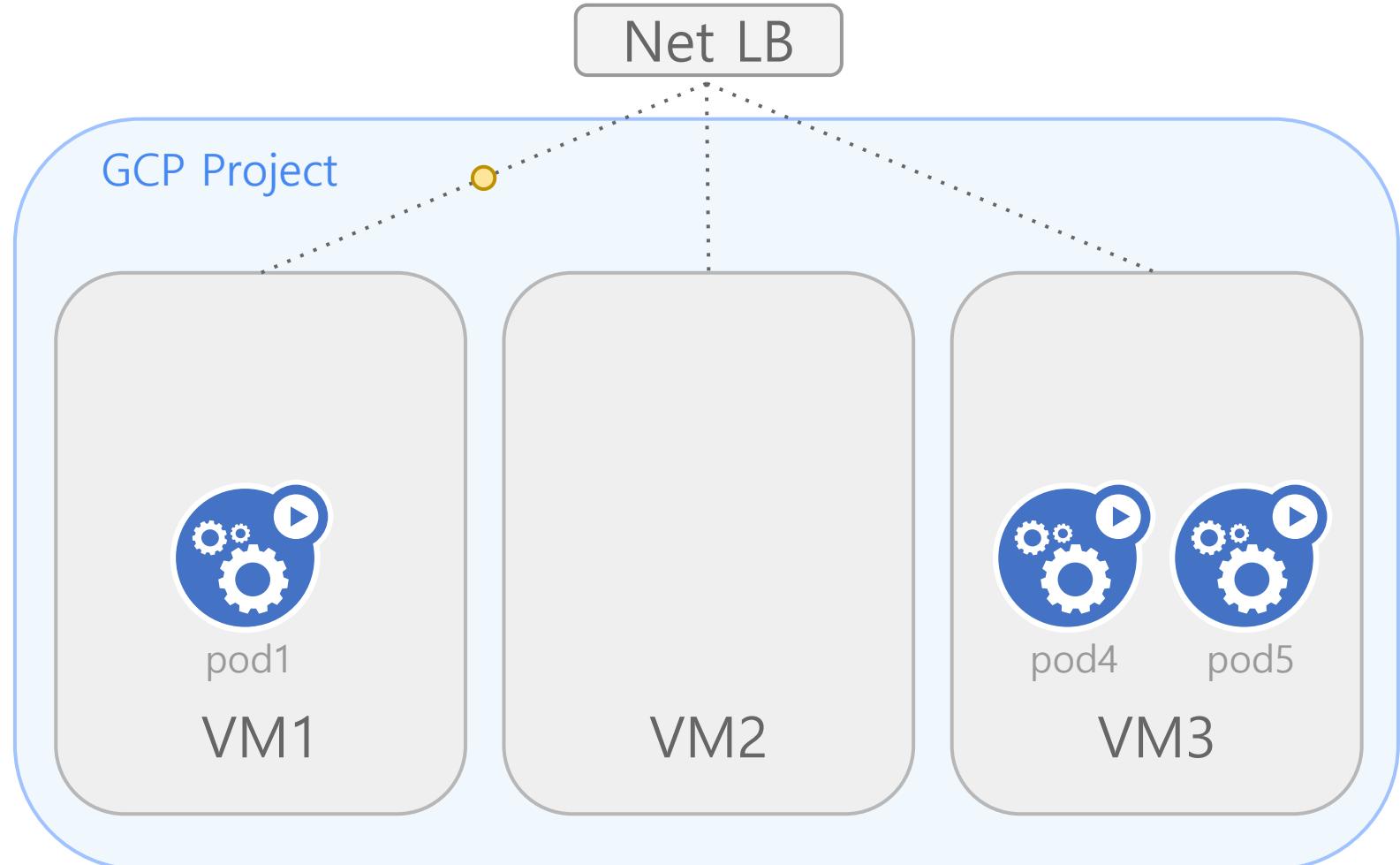
# Service type: LoadBalancer



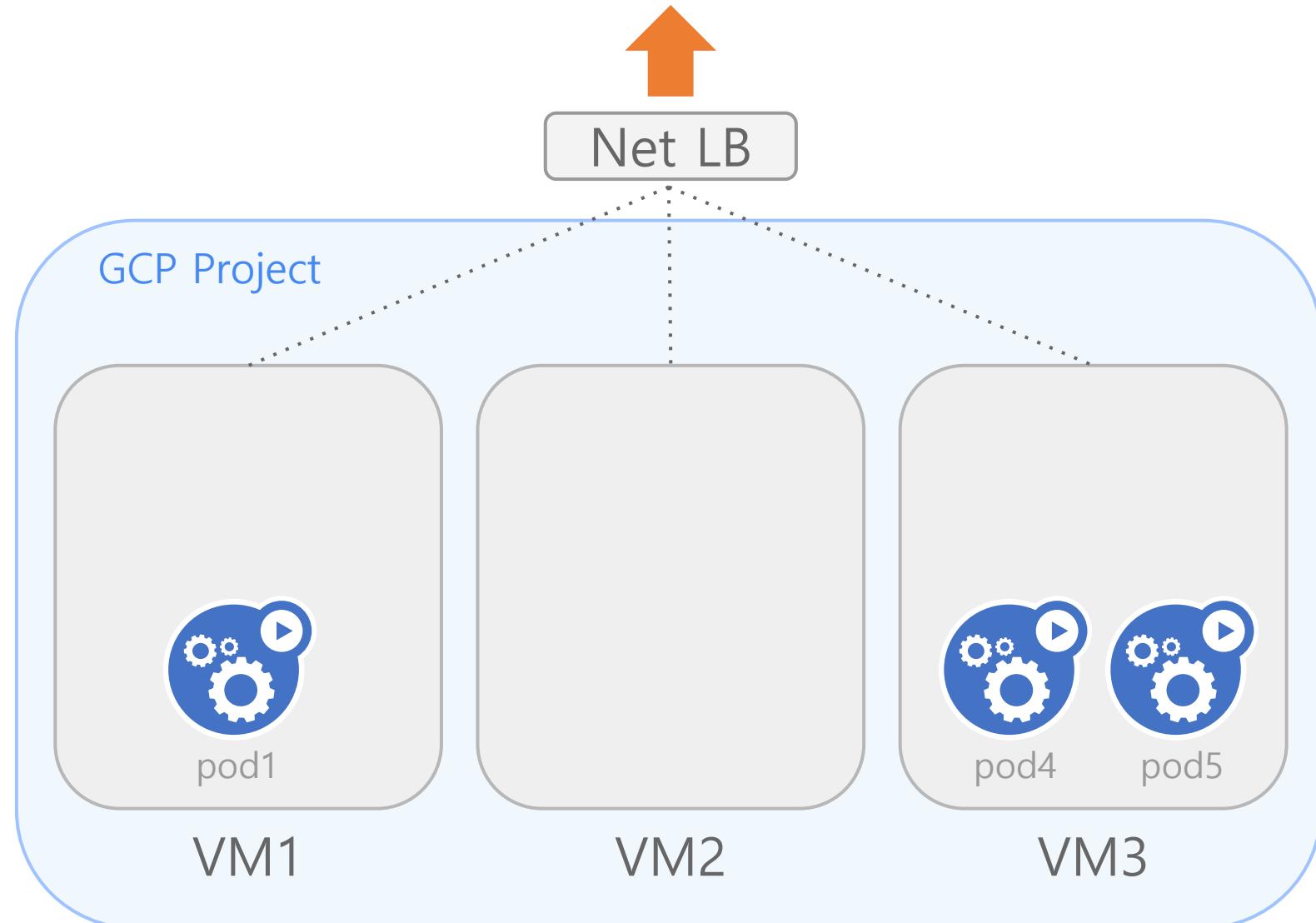
# Service type: LoadBalancer



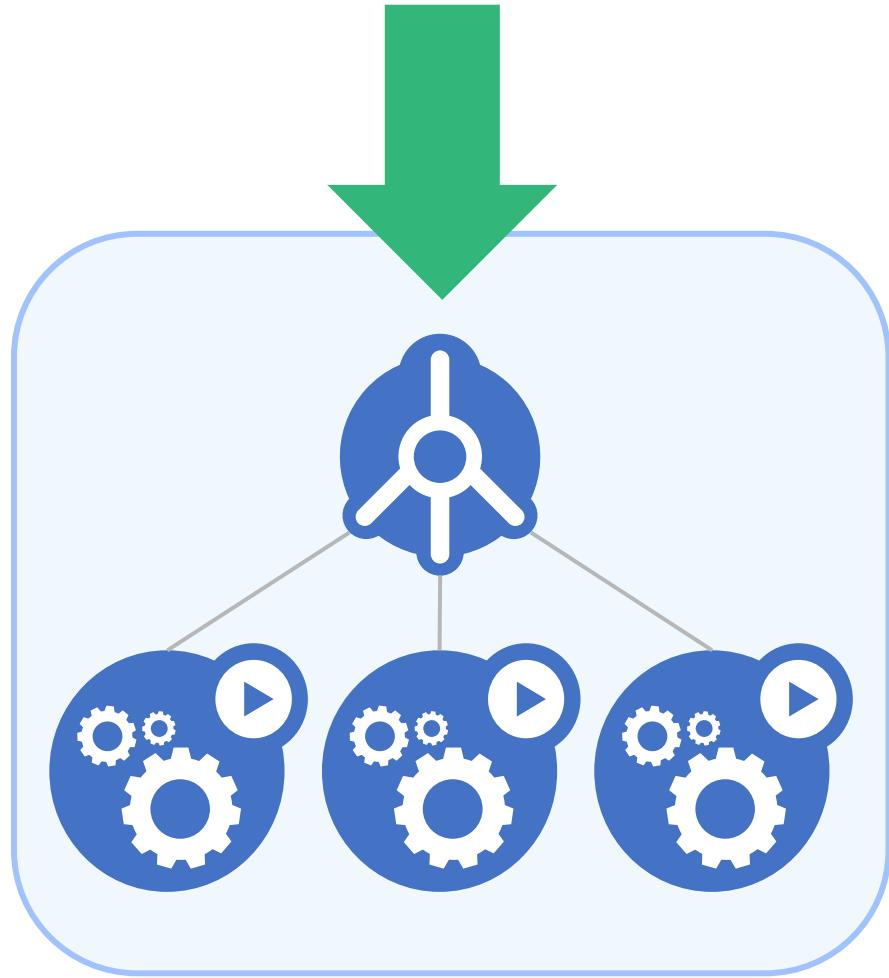
# Service type: LoadBalancer



# Service type: LoadBalancer



# Service type: LoadBalancer



# **Networking Resources**

- Ingress**

# Ingress

Many apps are HTTP/HTTPS

Services are L4 (IP + port)

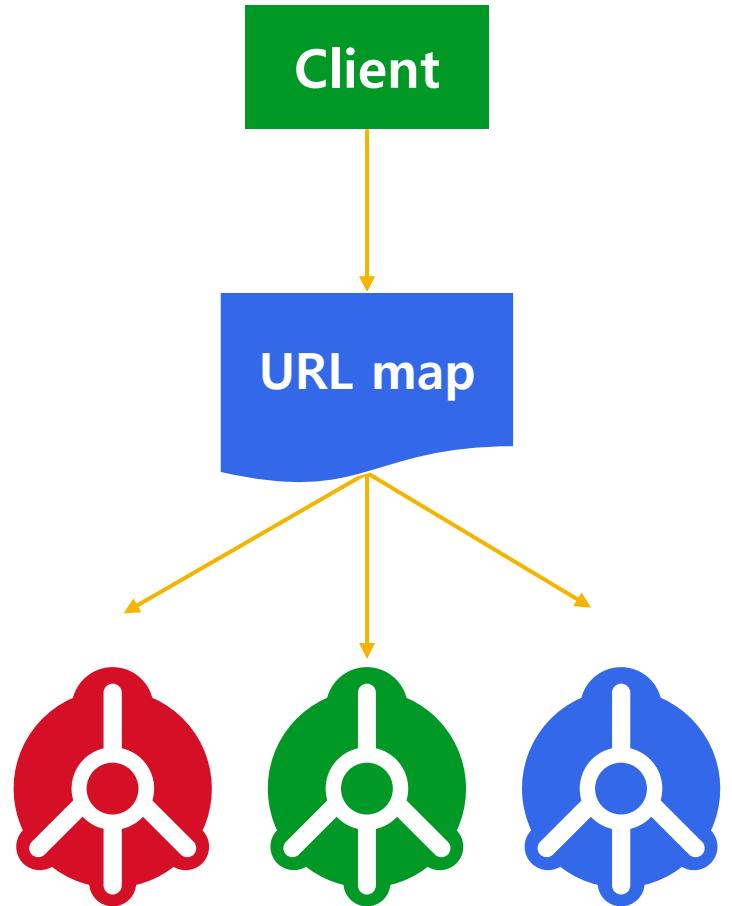
Ingress maps incoming traffic to backend services

- By HTTP host headers
- By HTTP URL paths

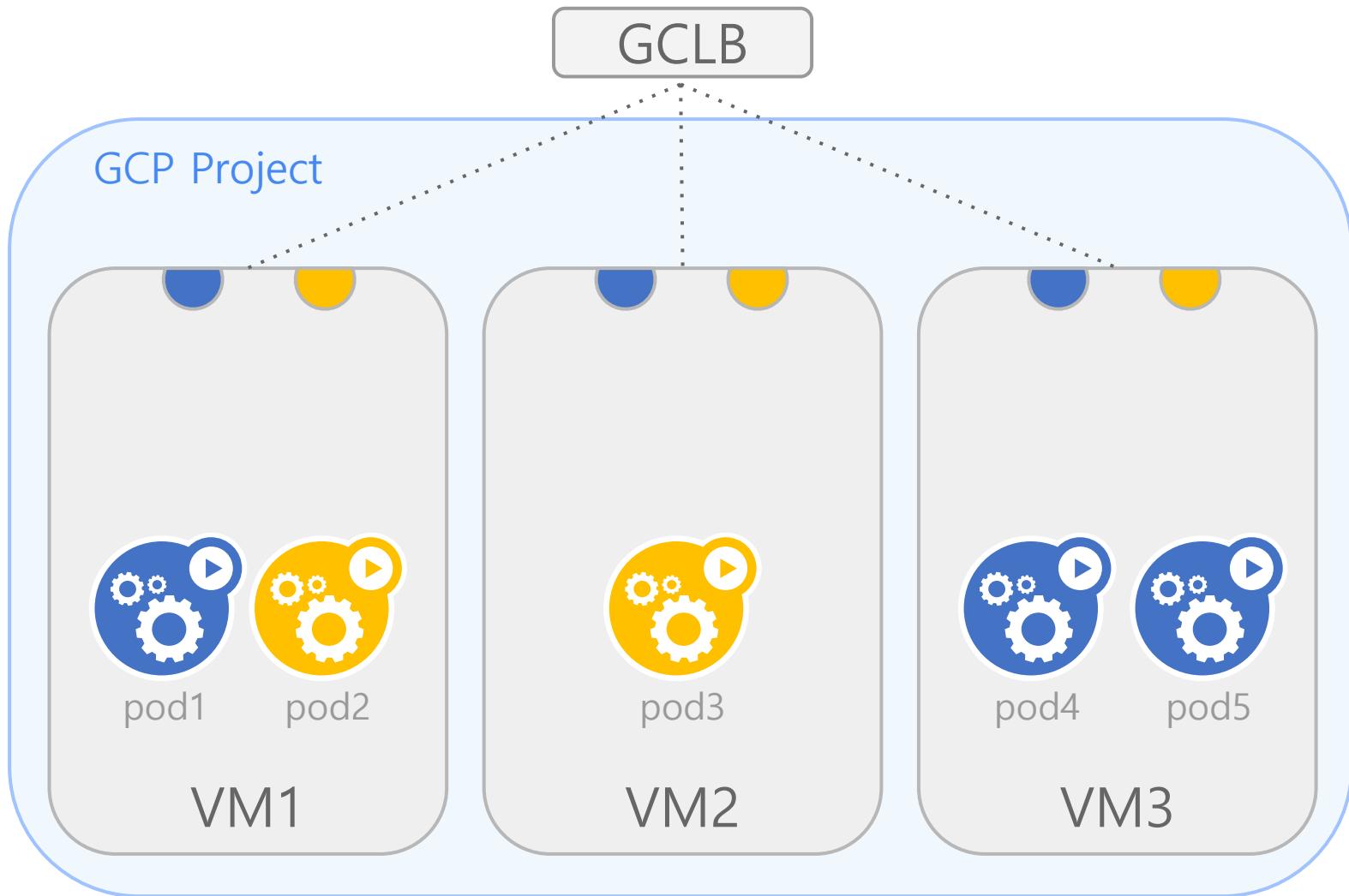
Work in conjunction with Services

Handled by Ingress controllers

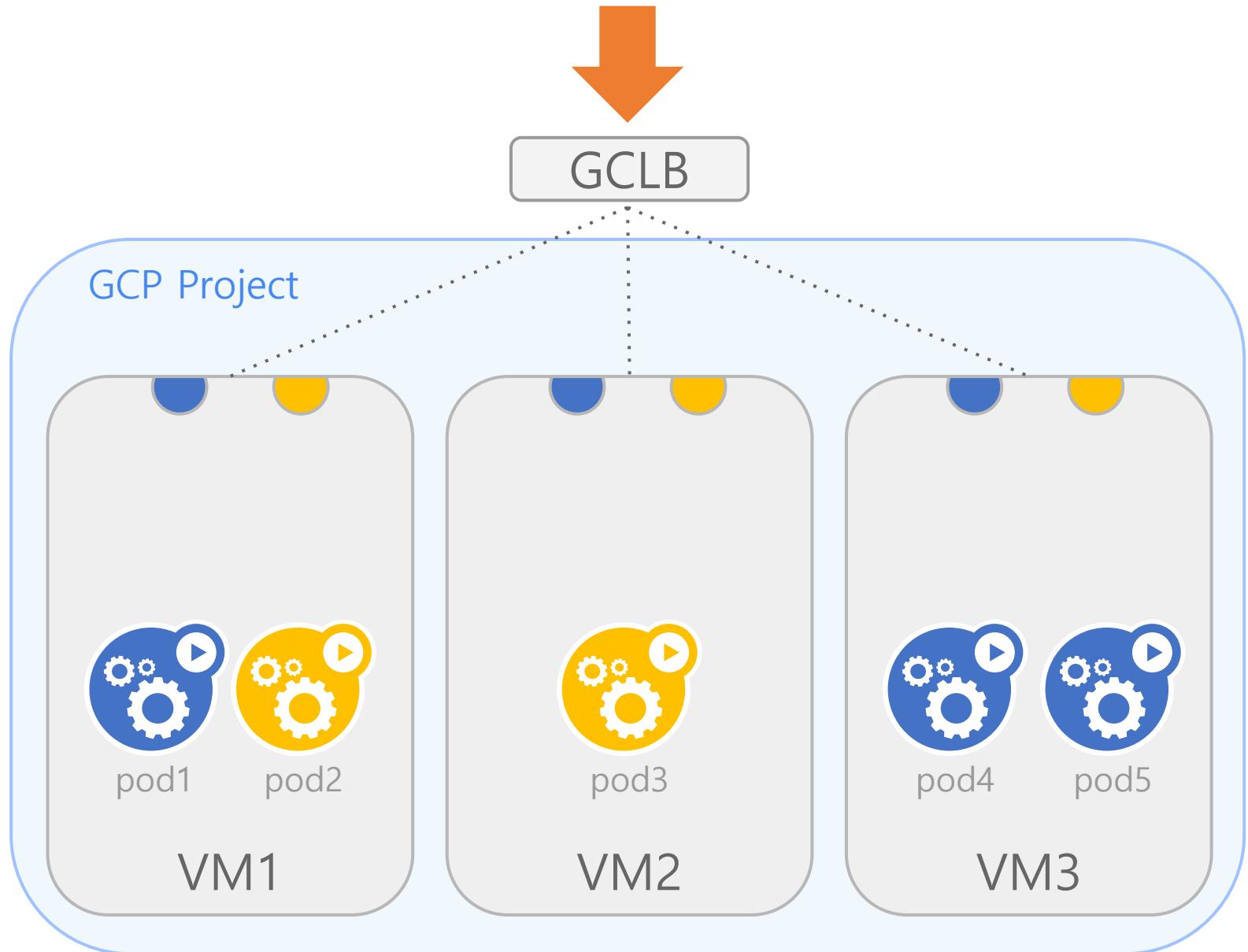
- GCE, NGINX, Traefik, and other implementations to choose from
- Controller has to be running within the cluster



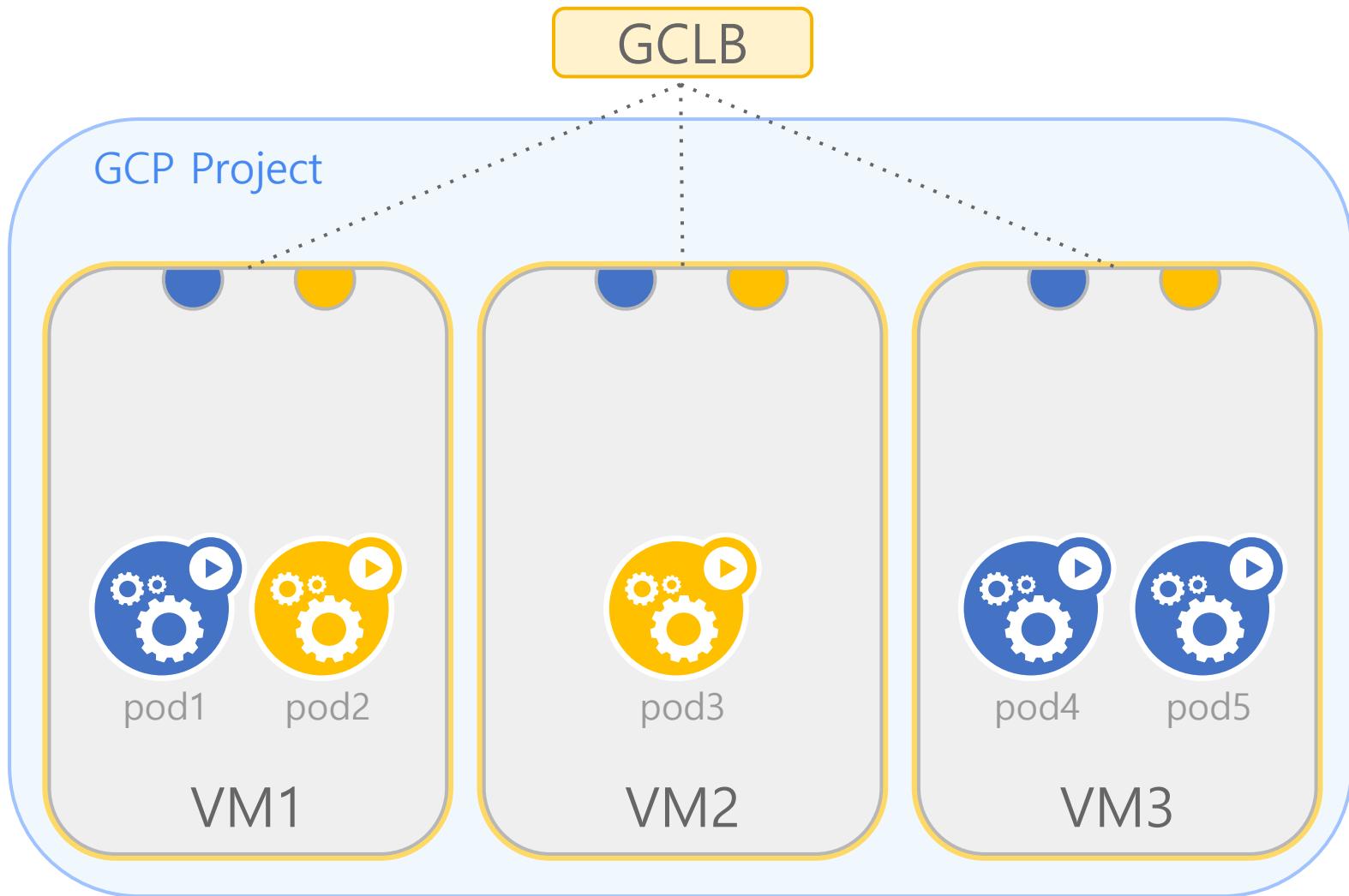
# Ingress



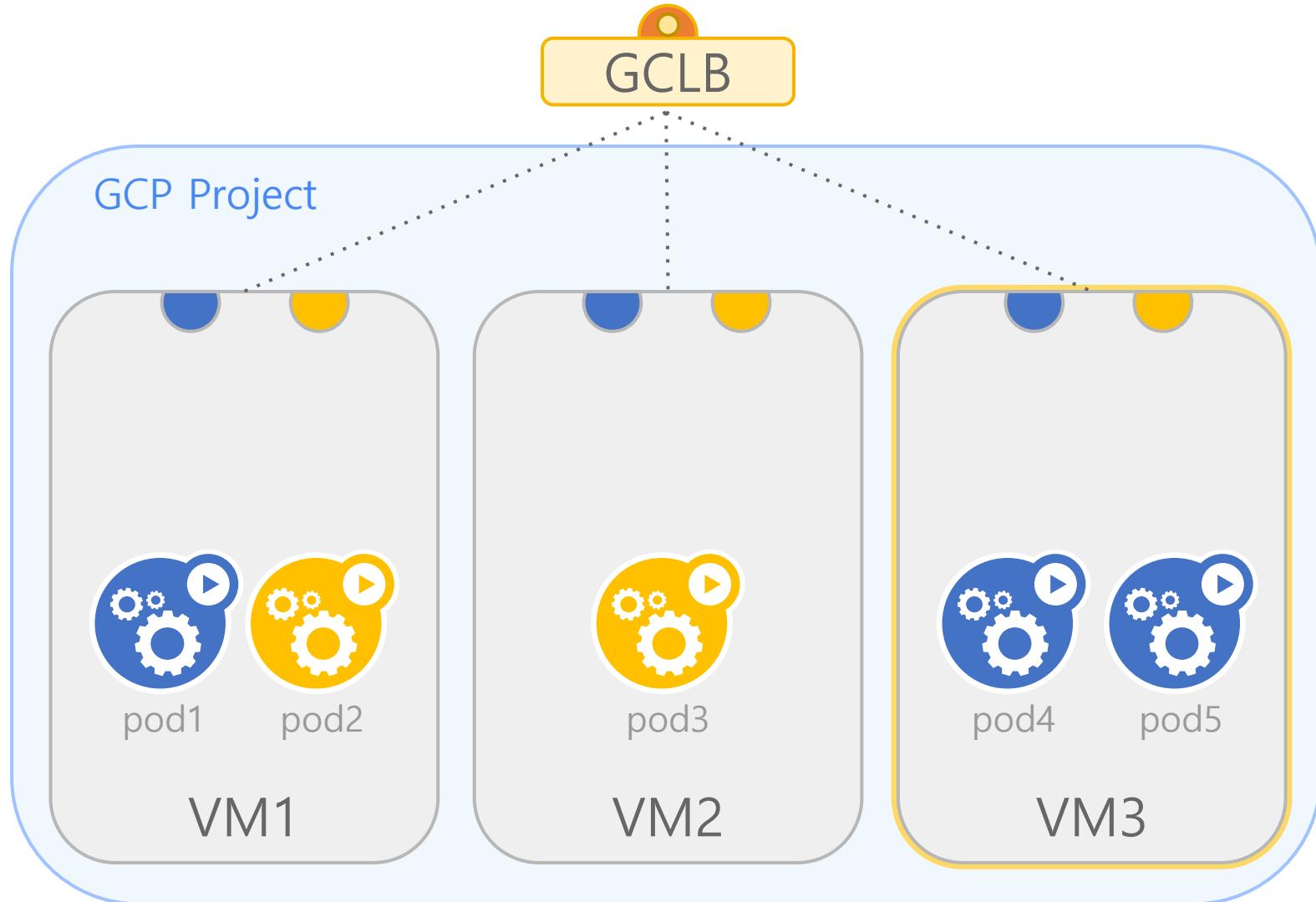
# Ingress



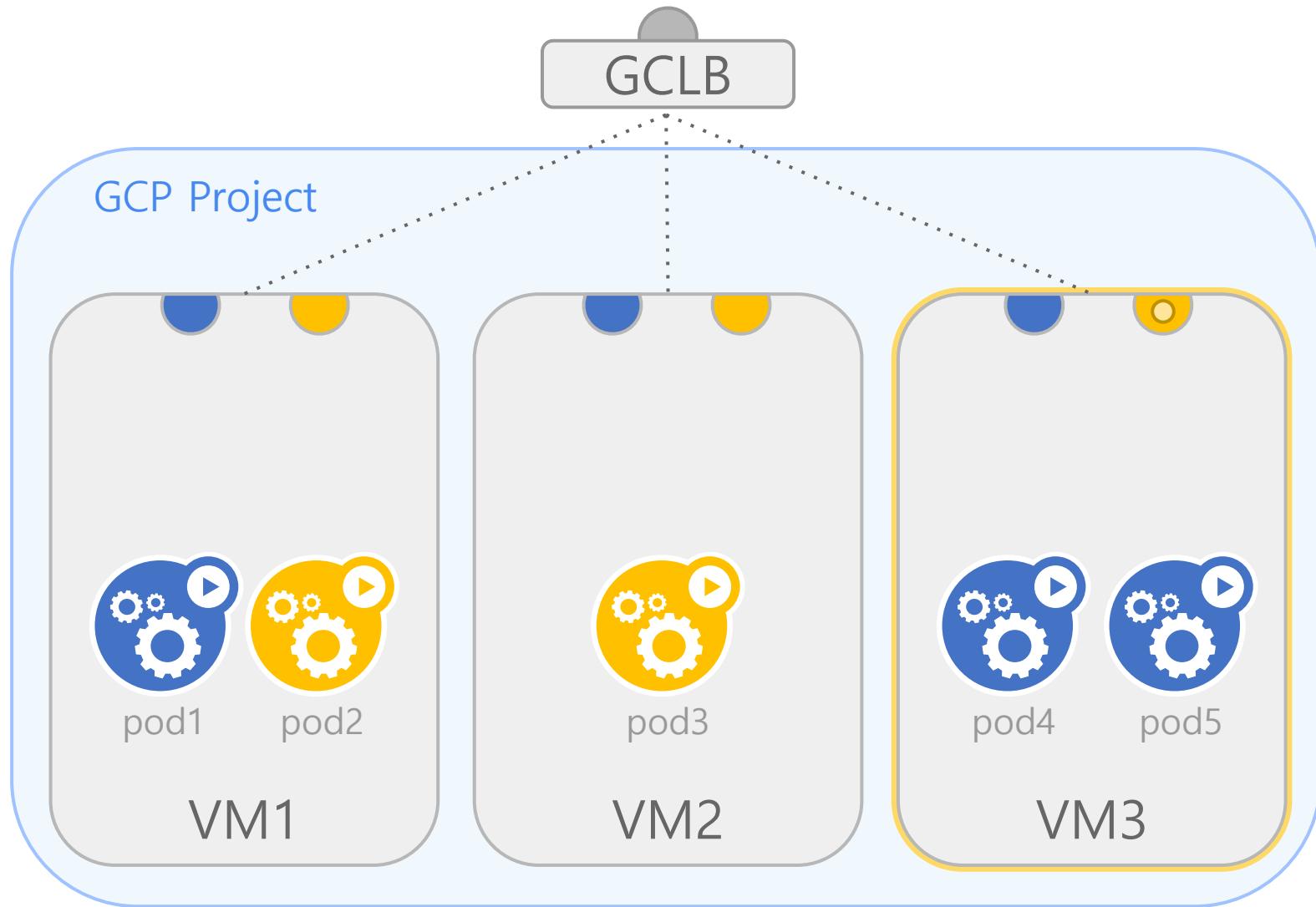
# Ingress



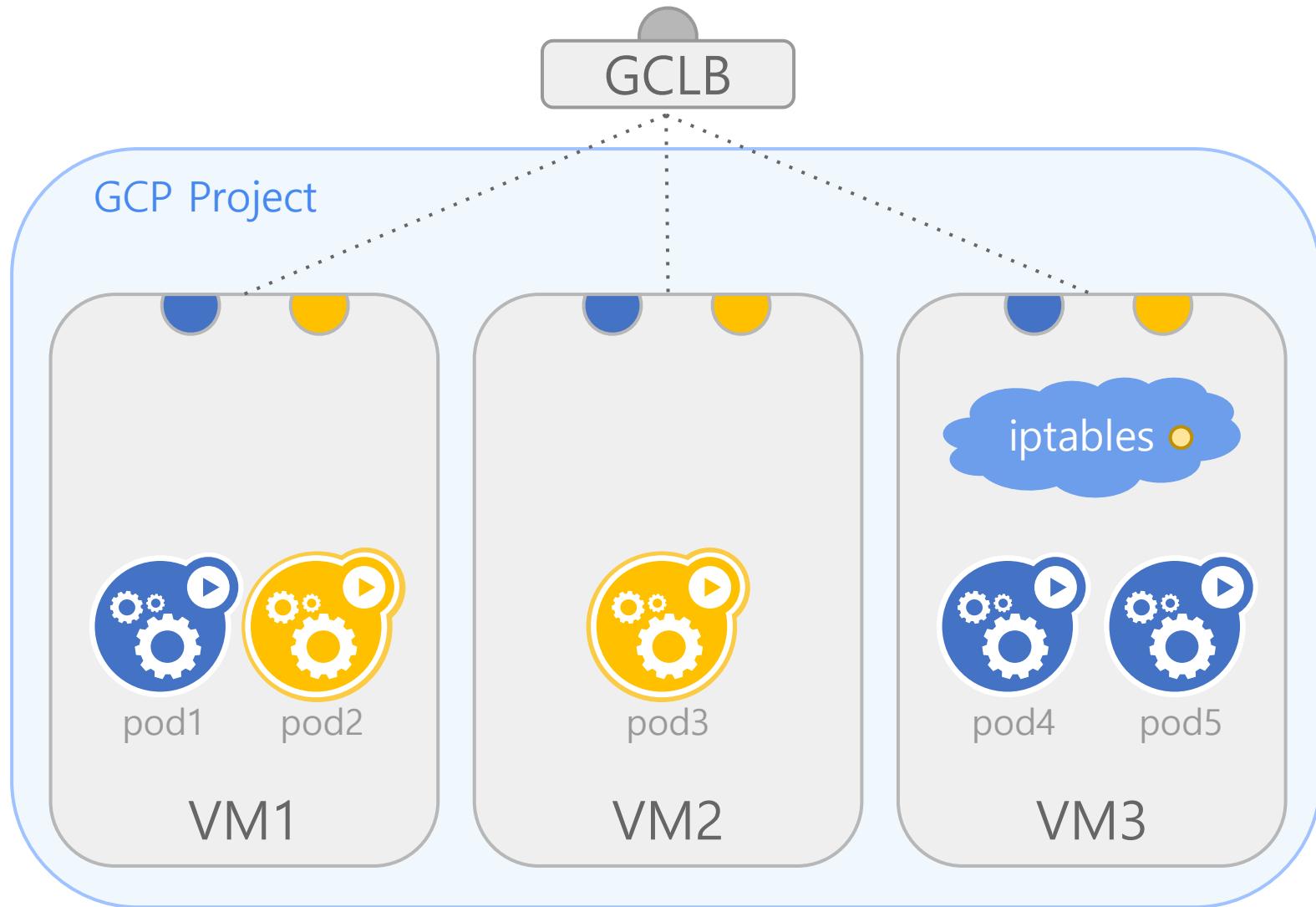
# Ingress



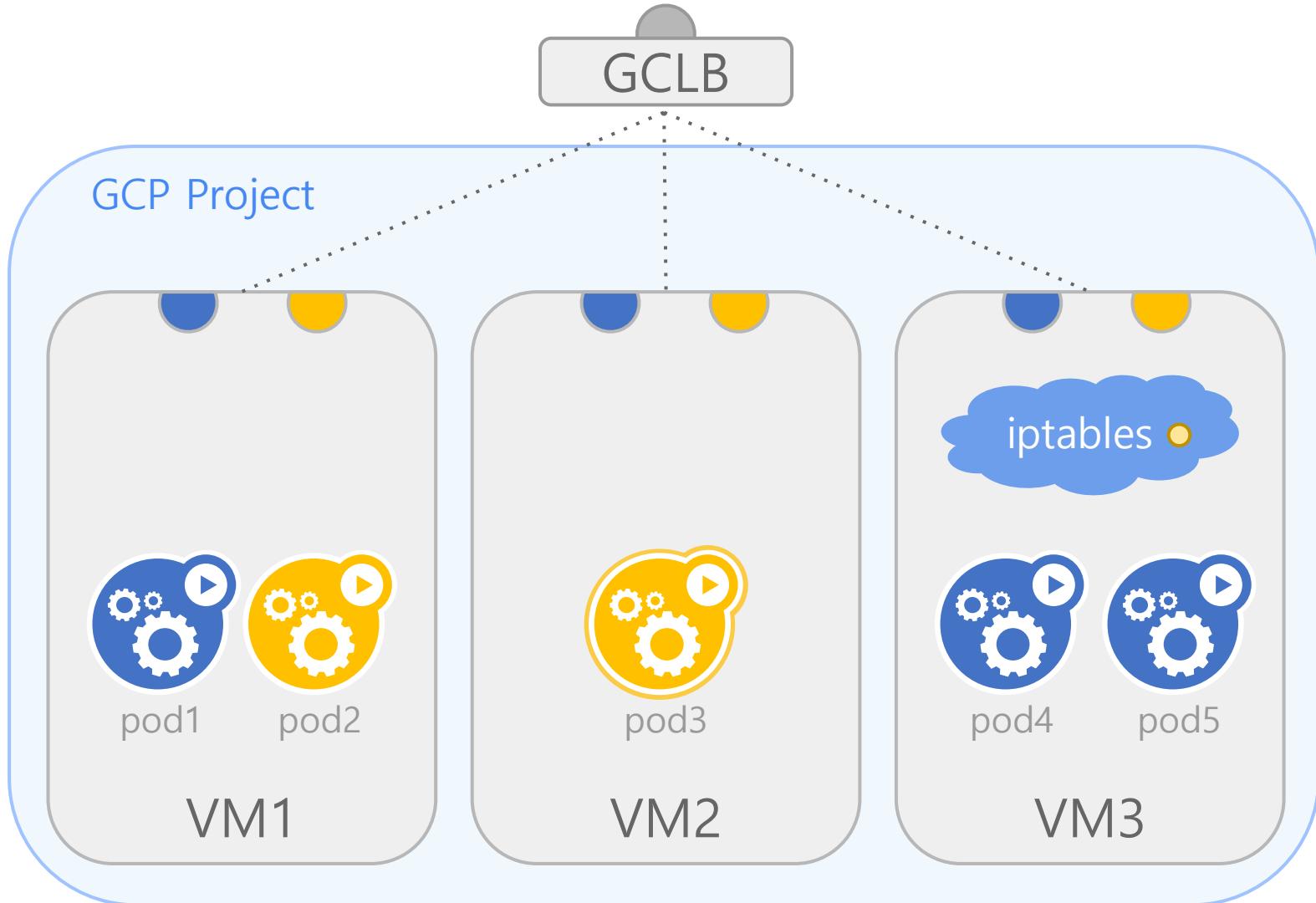
# Ingress



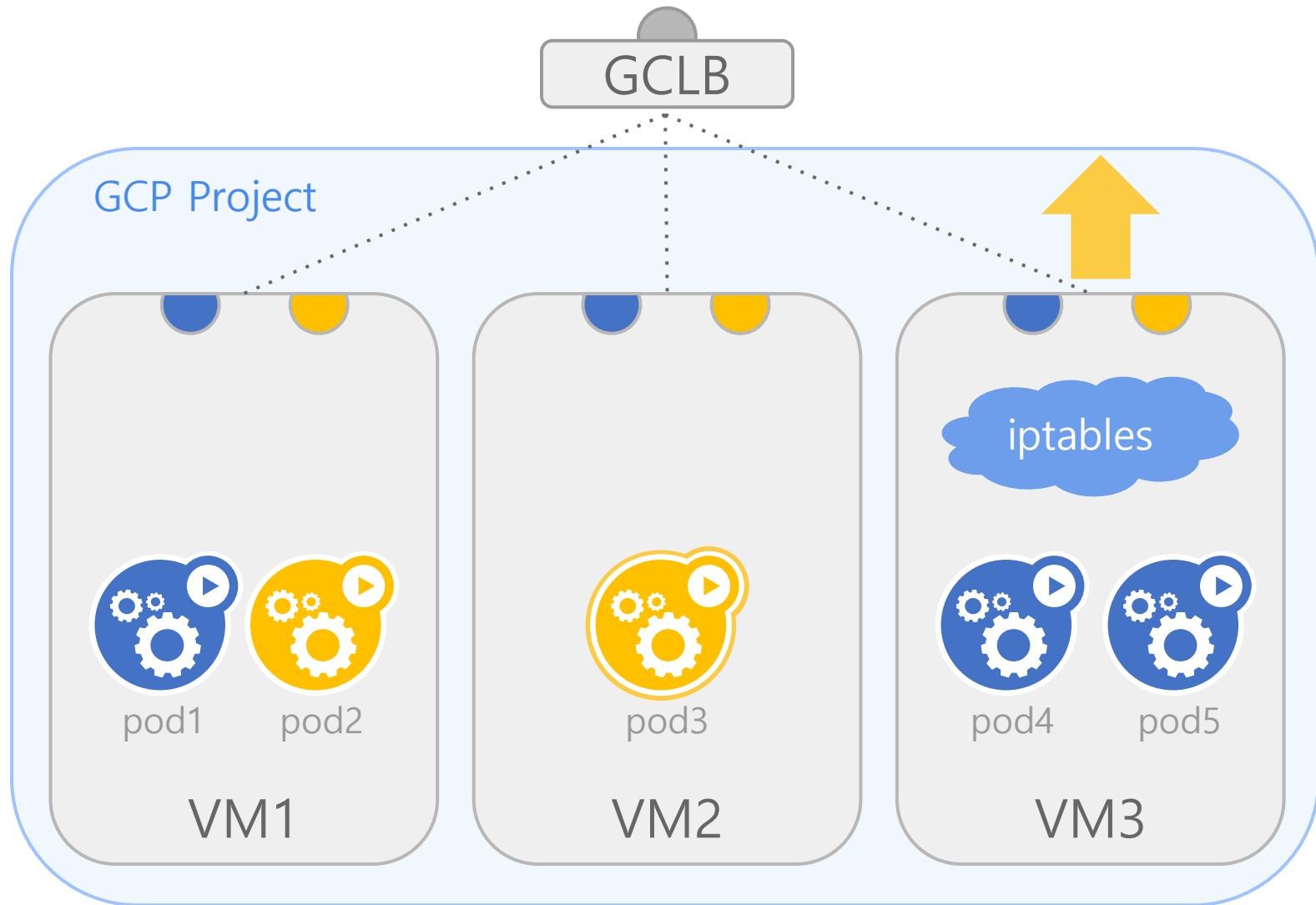
# Ingress



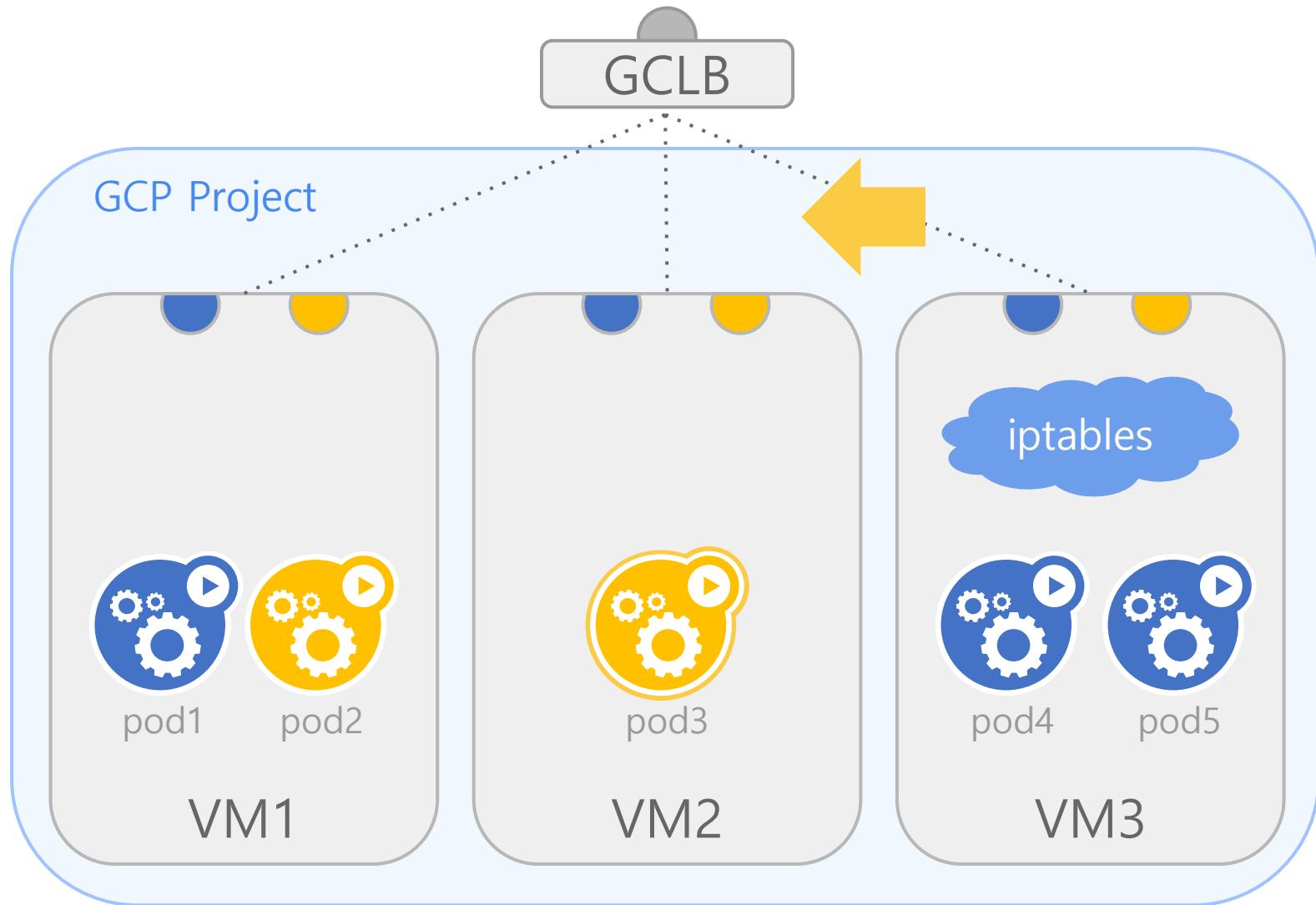
# Ingress



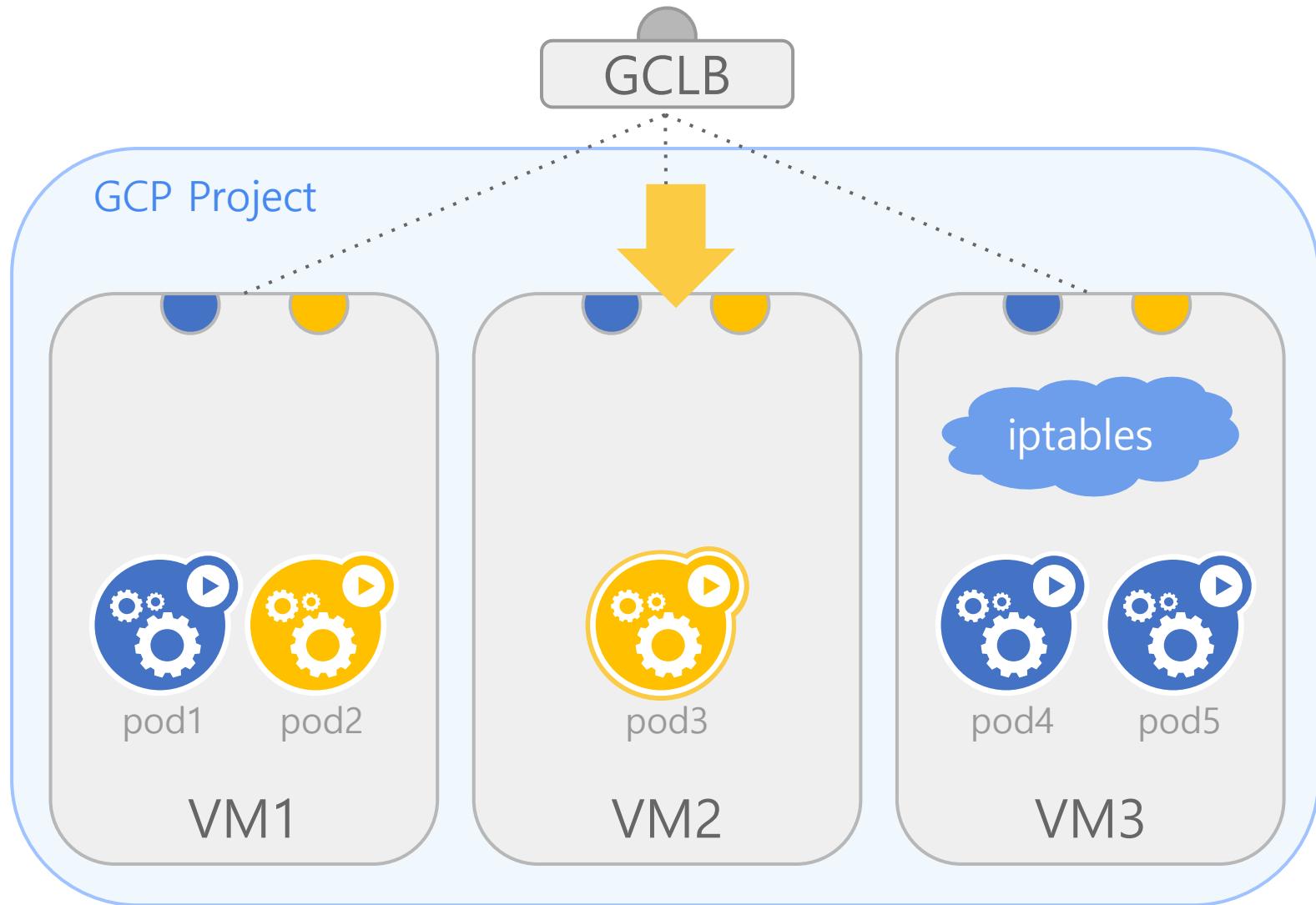
# Ingress



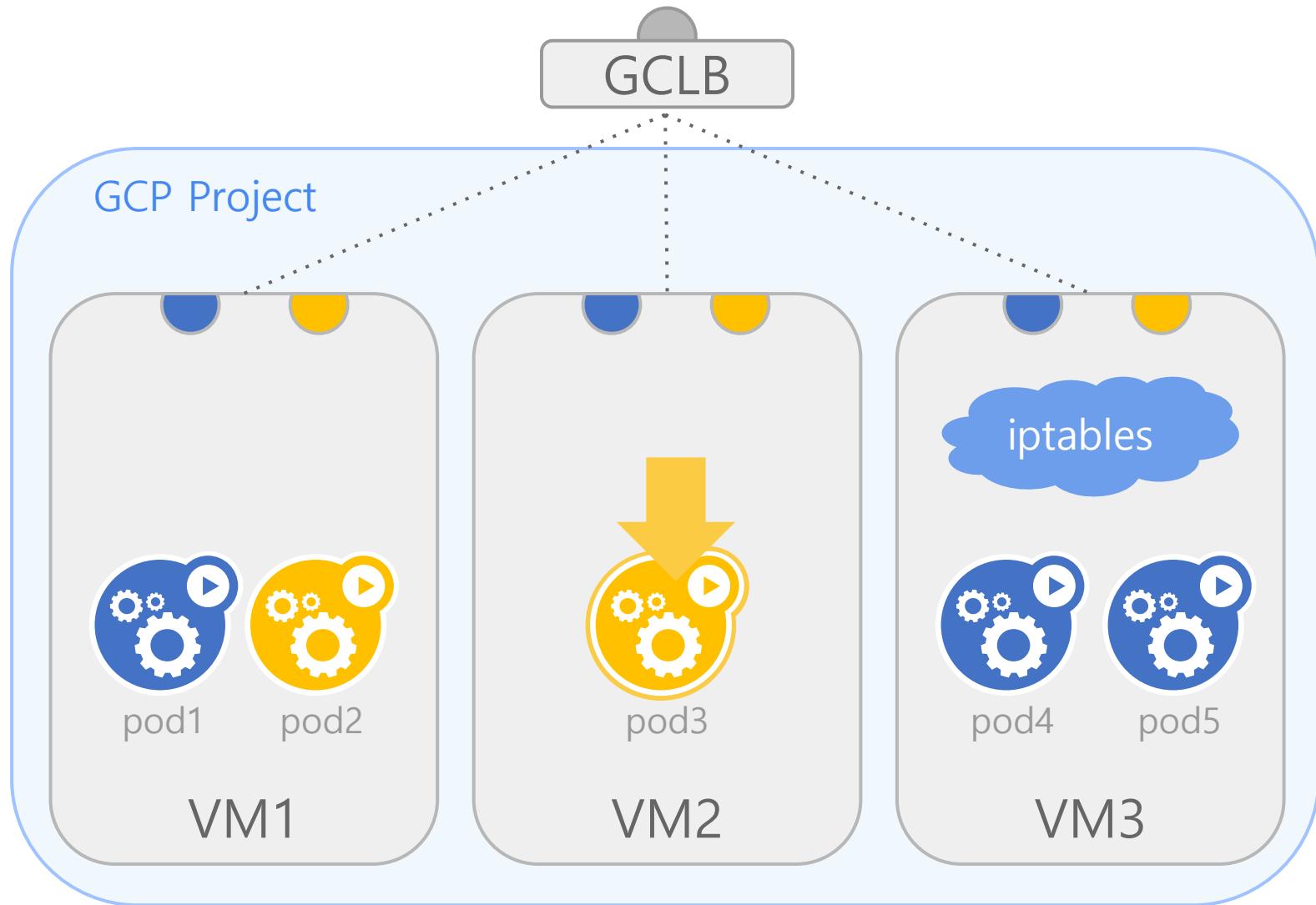
# Ingress



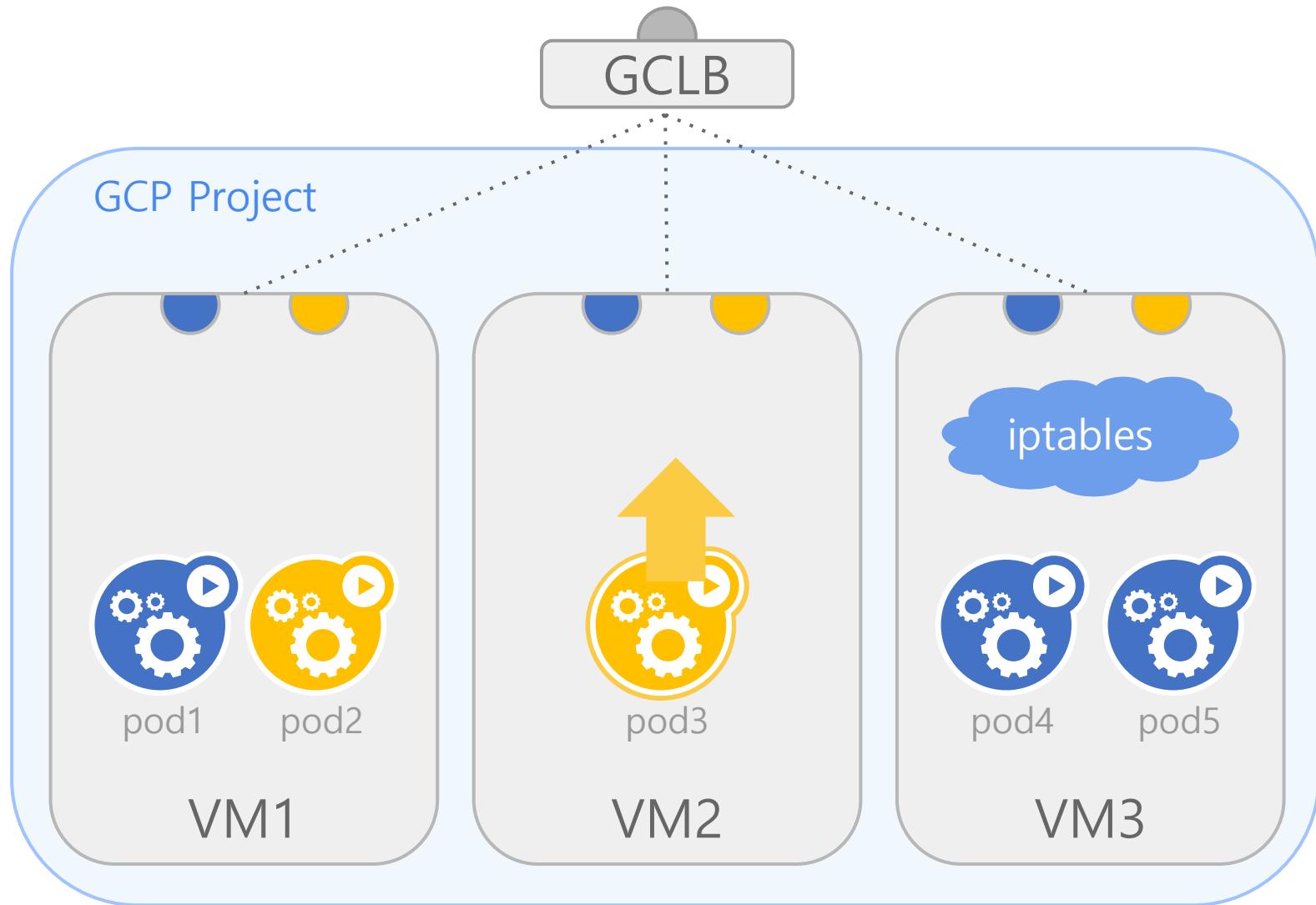
# Ingress



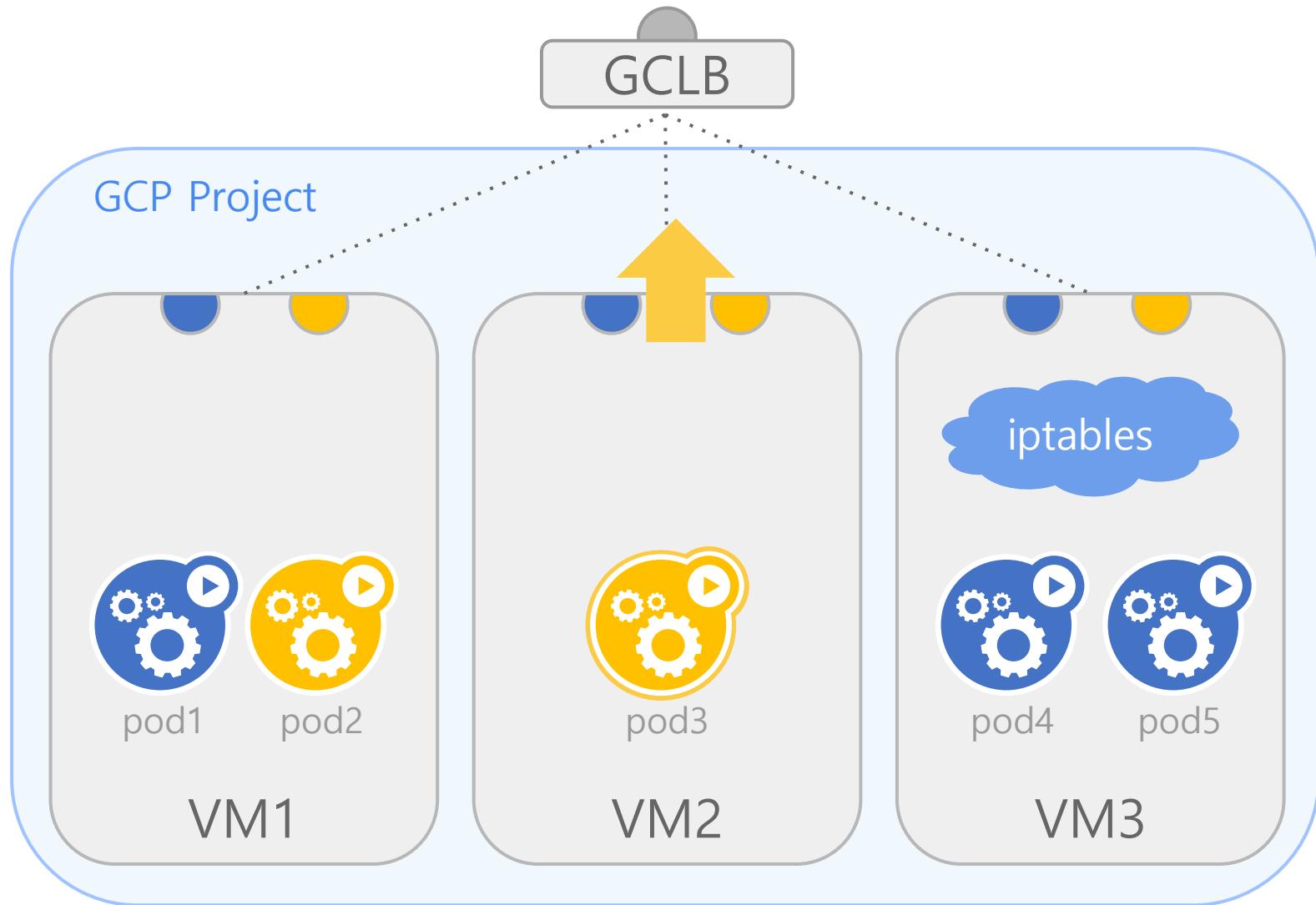
# Ingress



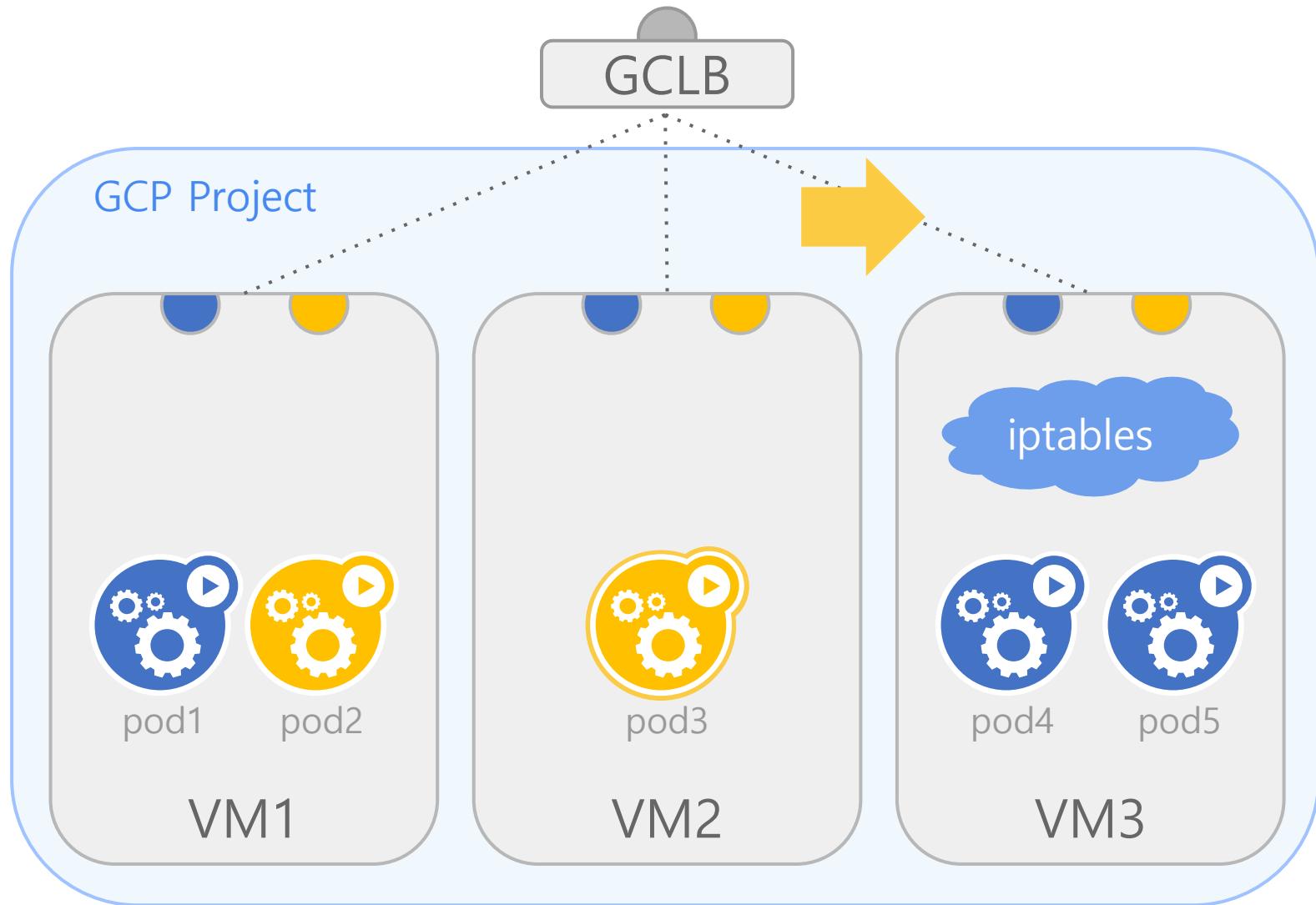
# Ingress



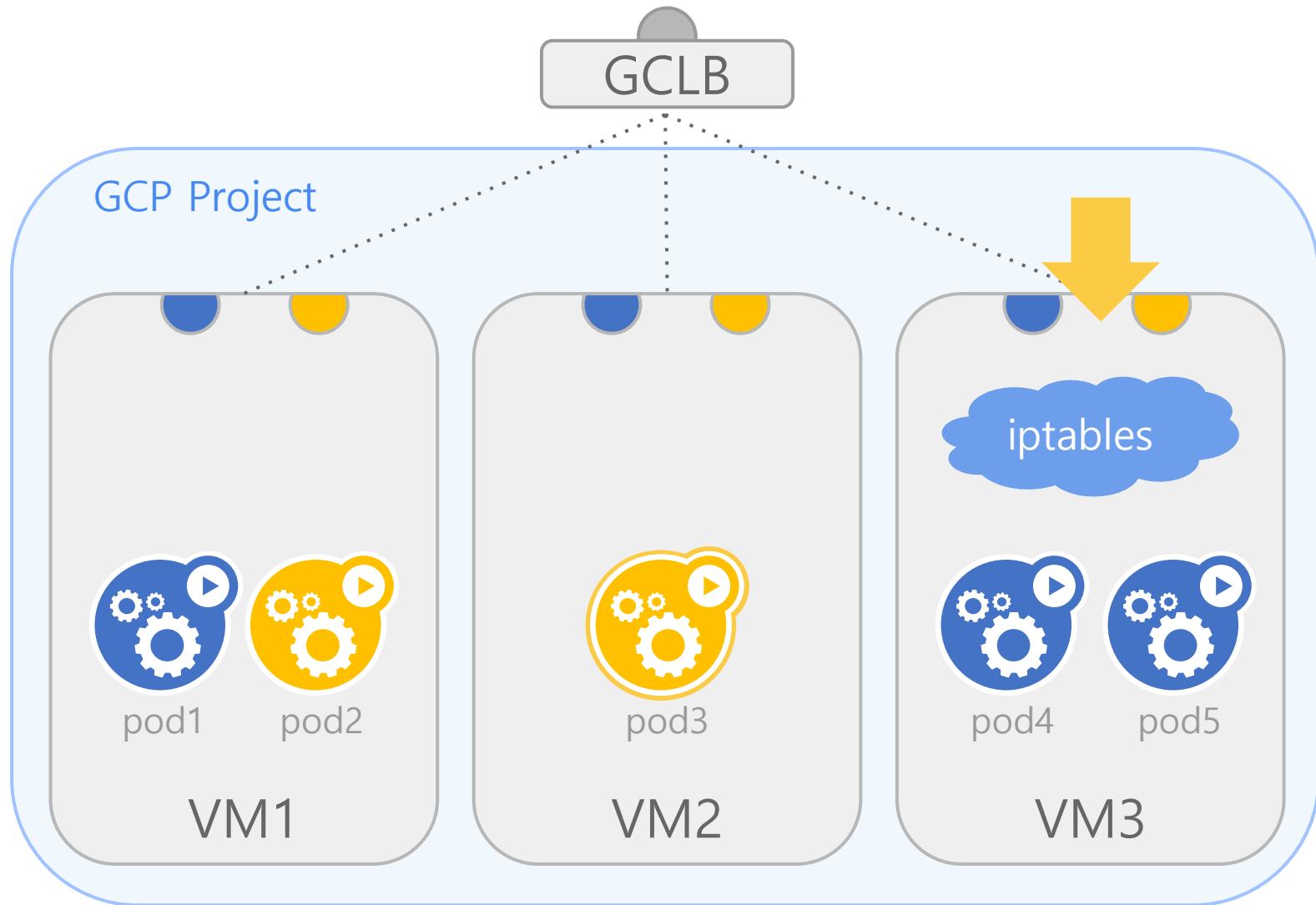
# Ingress



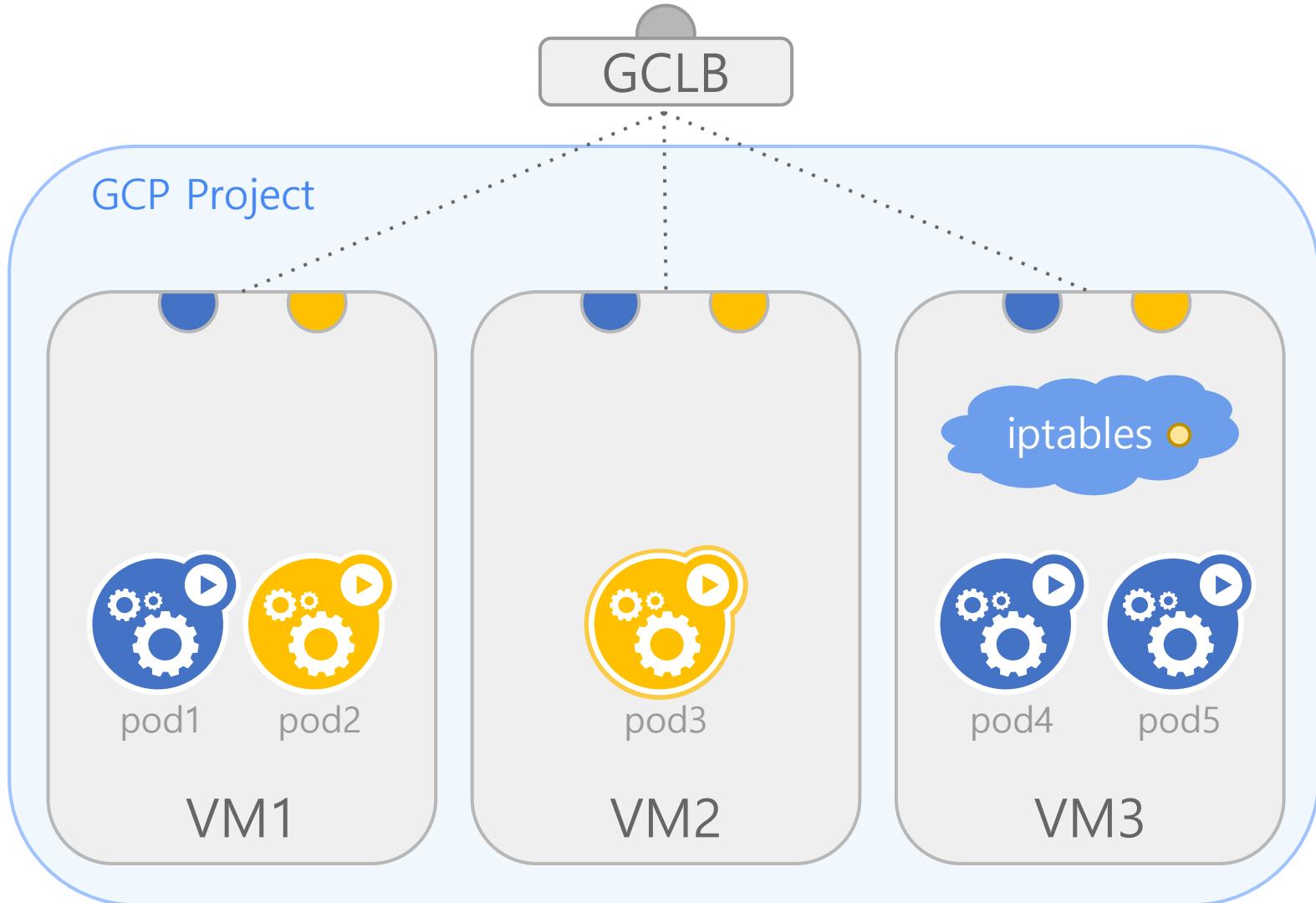
# Ingress



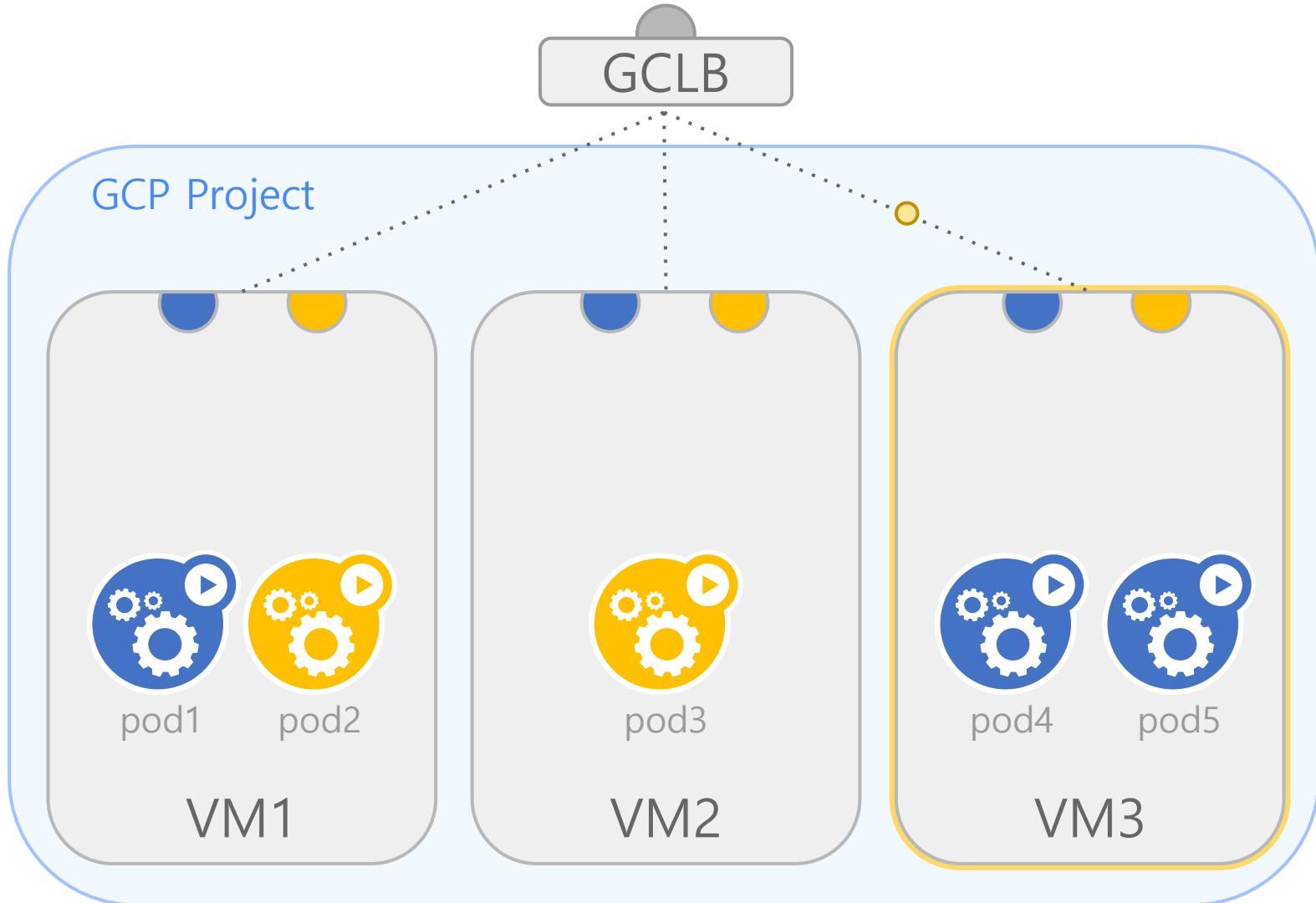
# Ingress



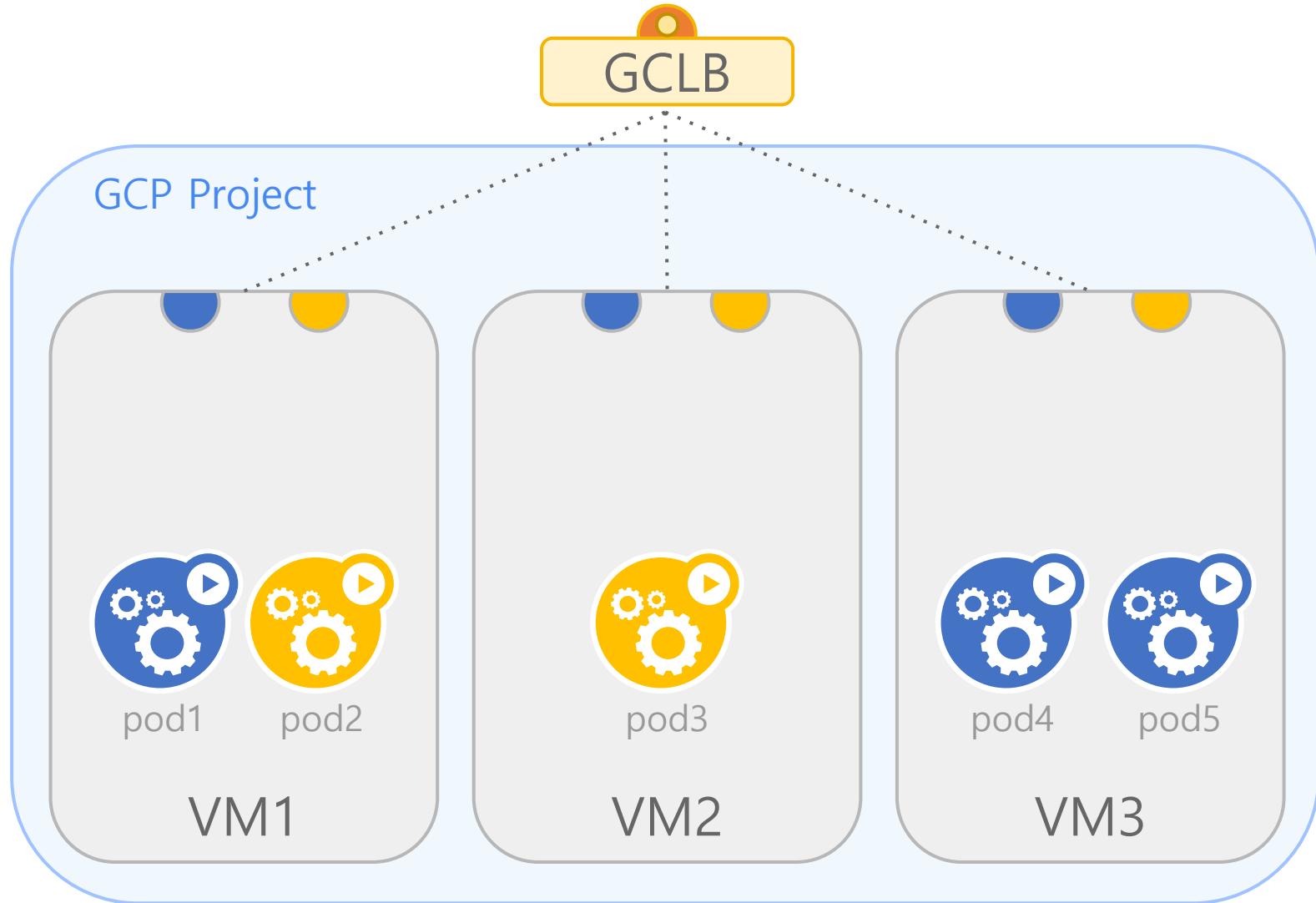
# Ingress



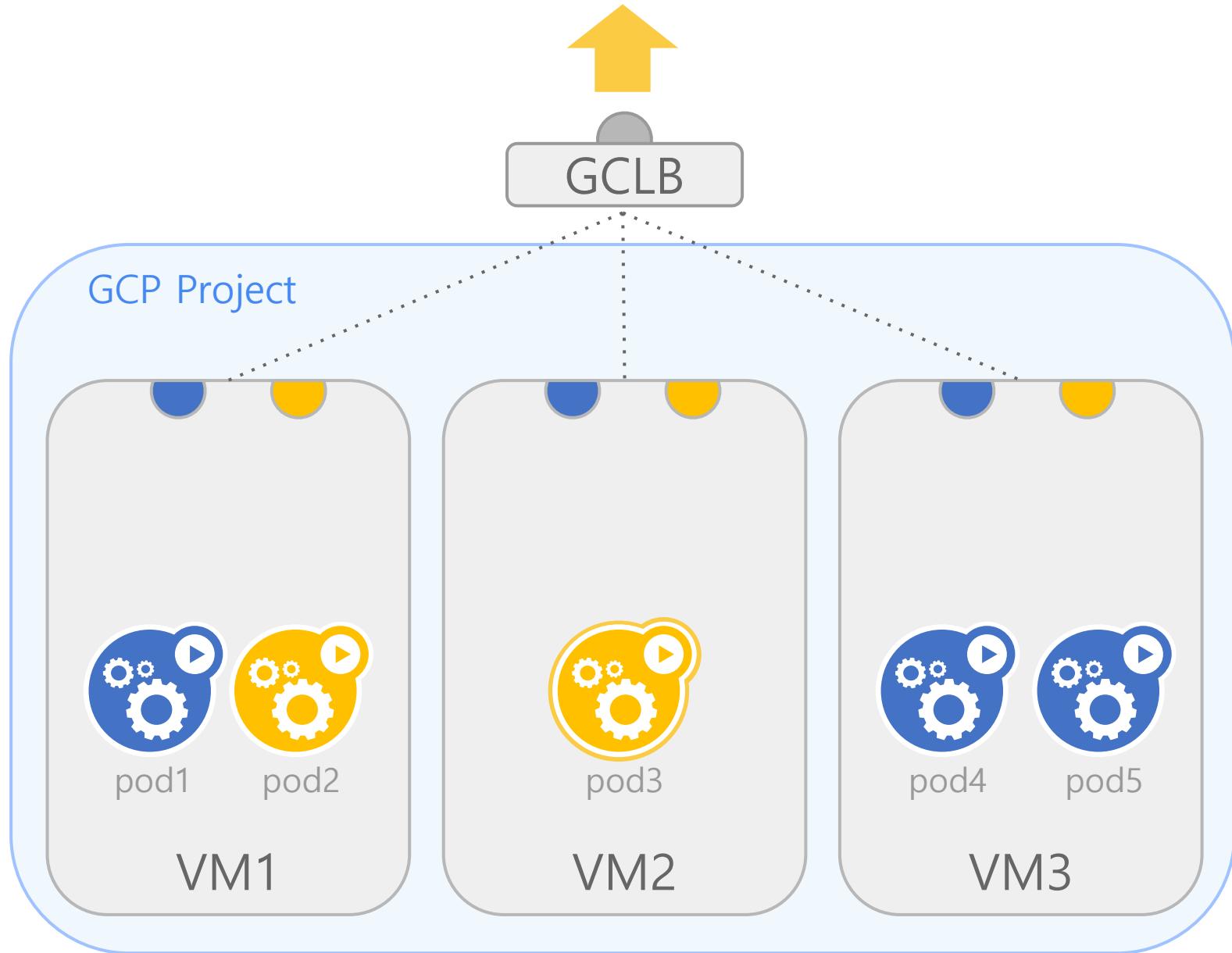
# Ingress



# Ingress



# Ingress



# **Networking Resources**

**- Comparison**

# Comparison

[More details here](#)

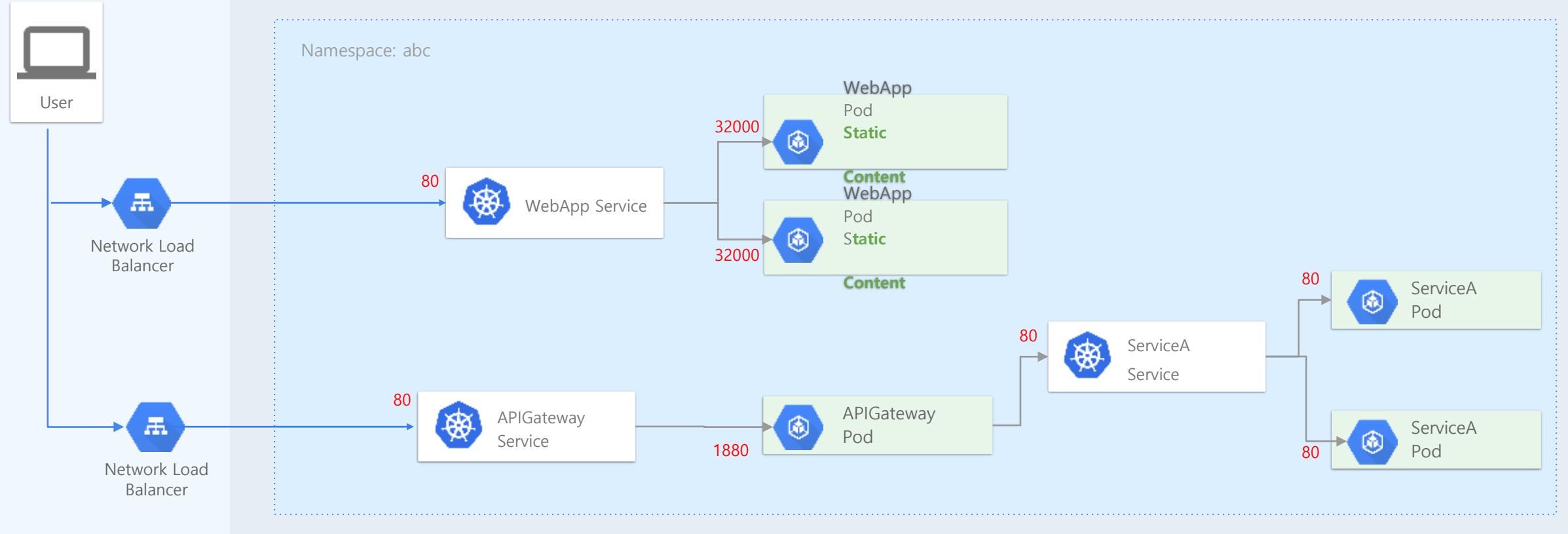
Feature	GCE Ingress	Service type:LoadBalancer
Type of load balancing	<a href="#">L7 HTTP/s Proxy</a>	<a href="#">L3/4 Network LB</a>
Health checks set up automatically by Kubernetes Engine/K8s?	✓	✓
Usefulness of LB health checks in Kubernetes	Useful only to detect unreachable nodes. Users must still configure readiness/liveness probes.	
Access to the original client IP	✓ (through X-Forwarded- F or header; source IP will be LB)	✗ generally ✓ only when the <a href="#">externalTrafficPolicy field</a> equals Local.
Can route HTTP/s?	✓	✓
TLS termination	✓	✗
SNI	✓ Only ~10 certs per GCLB	✓ if you terminate SSL yourself
Sticky sessions (Session affinity)	✗ No	✓ Client IP
WebSockets	✓ Recently added	✓ (L3 doesn't distinguish)

# **Common Patterns**

# Via Service (type:LoadBalancer)

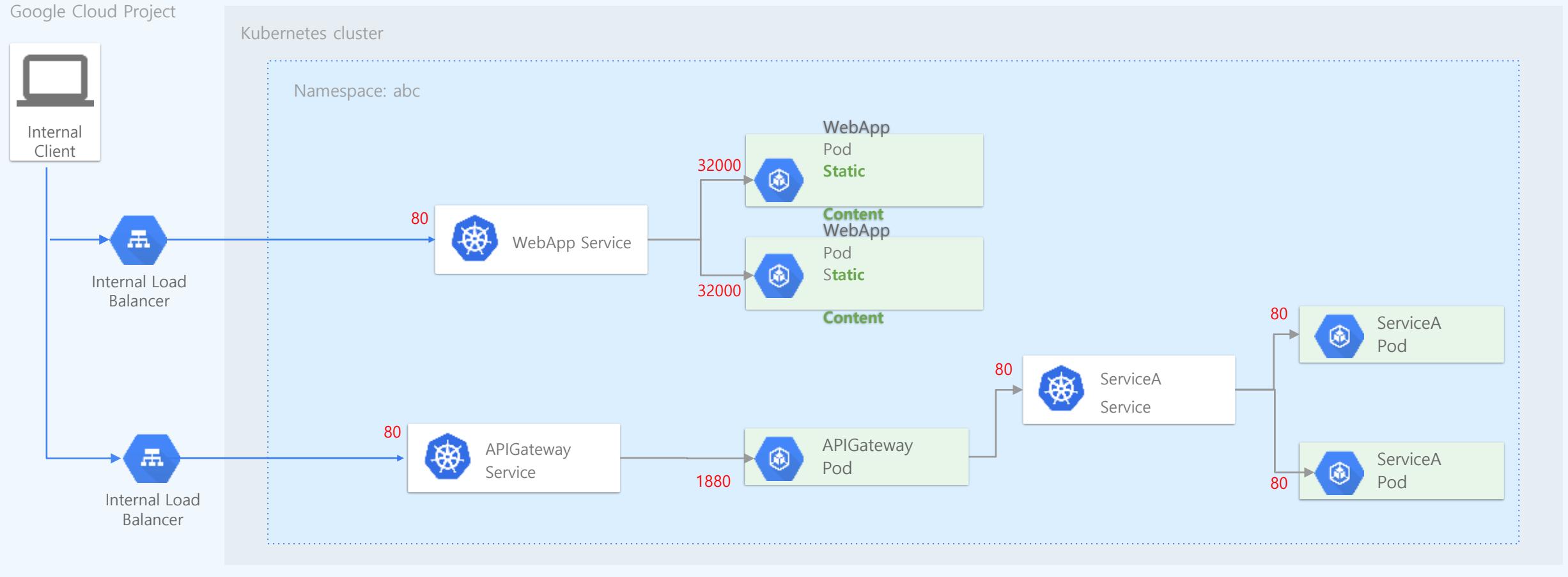
Kubernetes Engine Cluster

Google Cloud Project



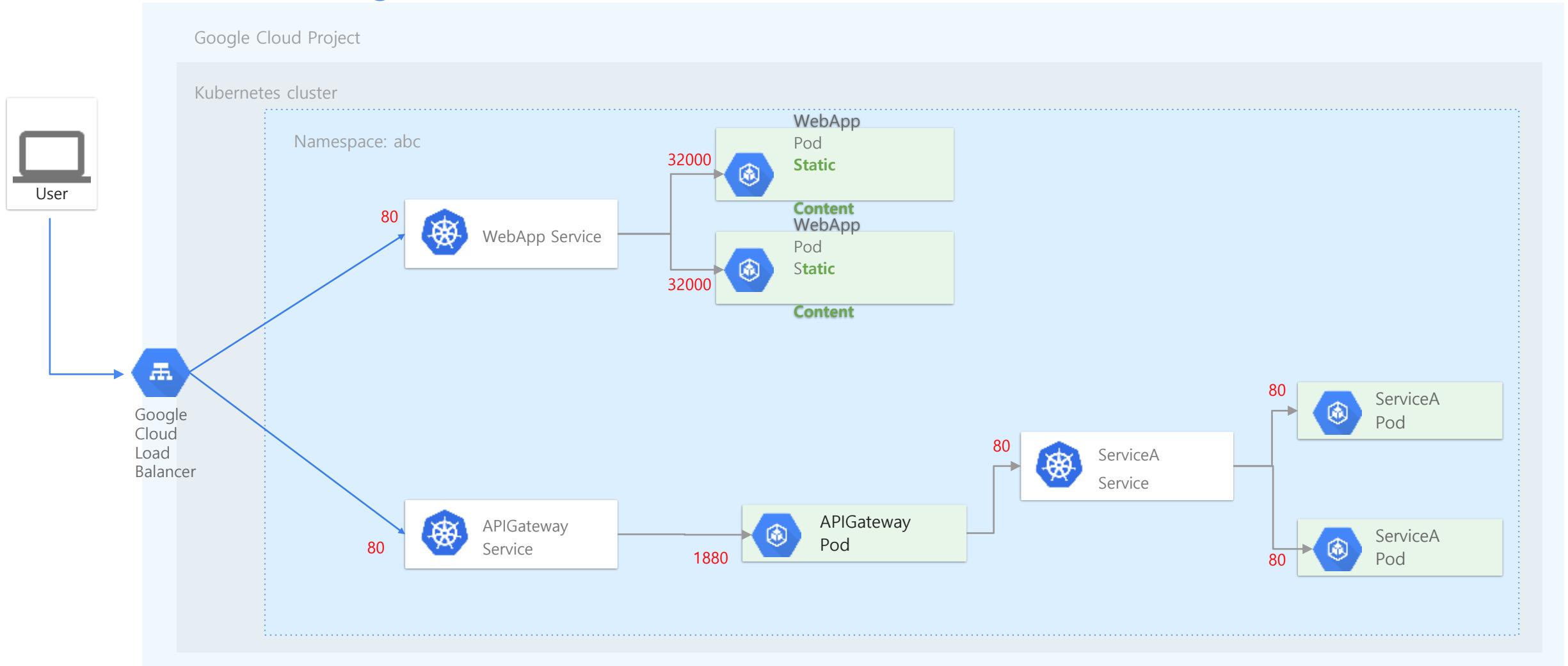
# Via Service (type:LoadBalancer) over ILB

## Kubernetes Engine Cluster



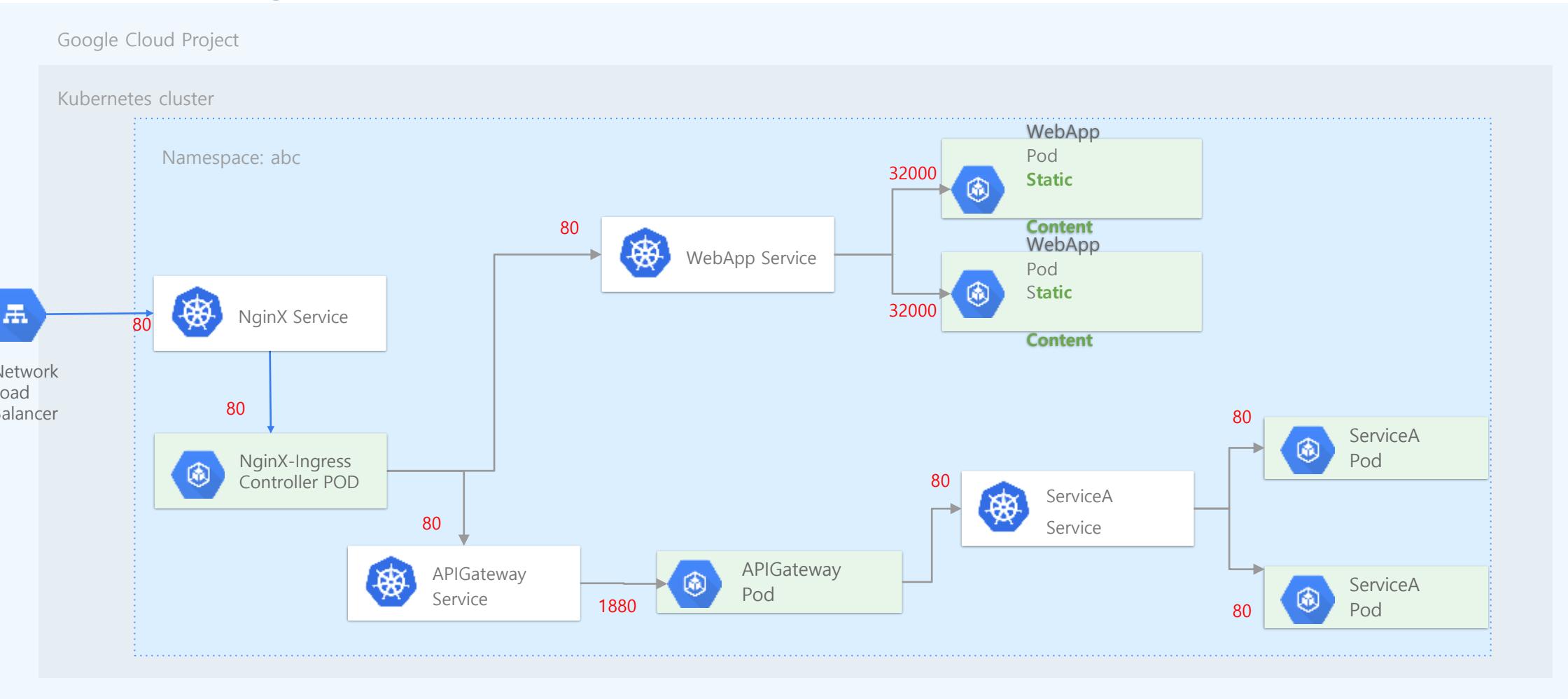
# Via Ingress (GCE Controller)

## Kubernetes Engine Cluster



# Ingress (via NGINX Ingress Controller)

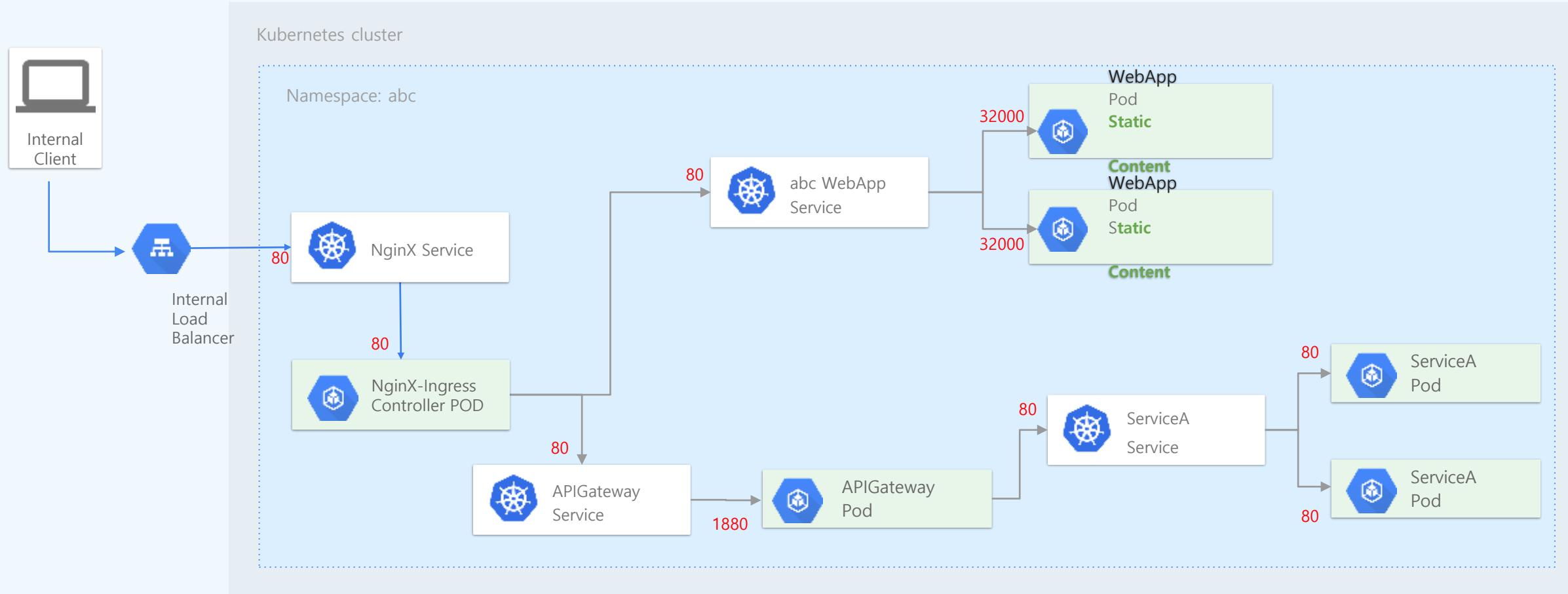
## Kubernetes Engine Cluster



# Ingress exposed via Service (type:LoadBalancer) over ILB

Google Cloud Project

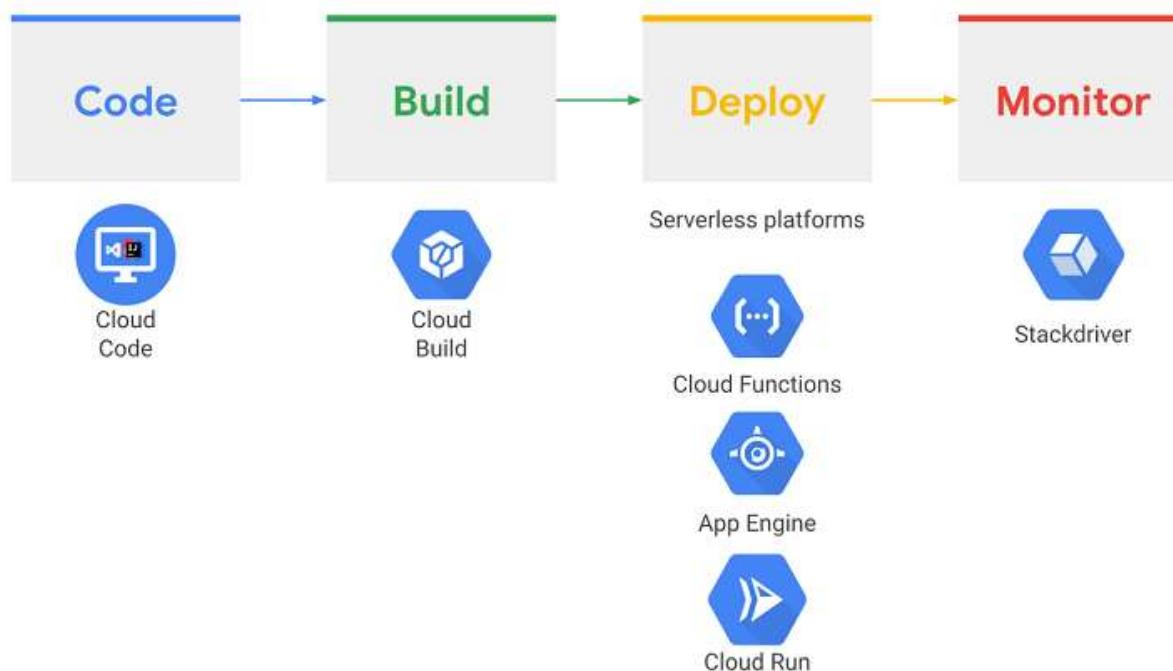
## Kubernetes Engine Cluster



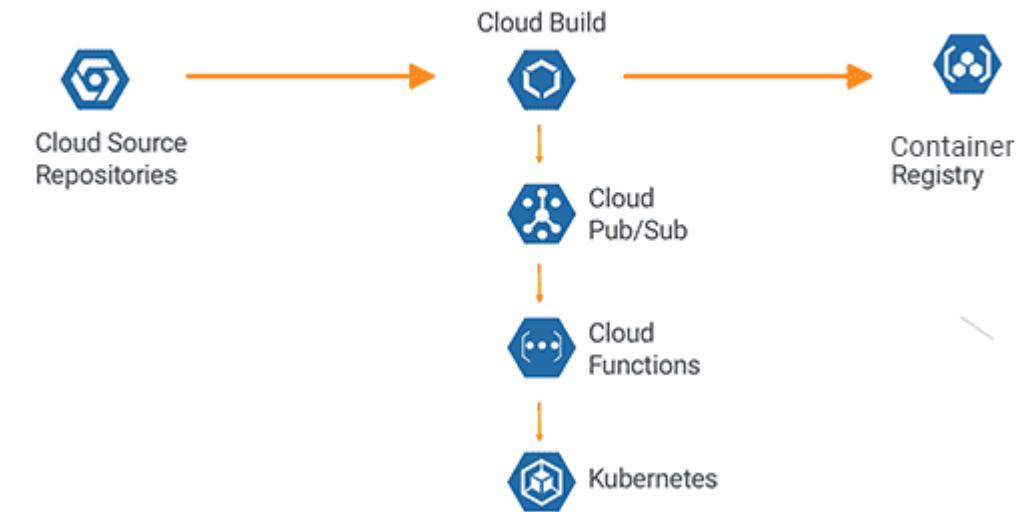
# Google의 Kubernetes 확장 전략

# 각 업체별 전략 – Google GCP

- CI/CD DevOps Pipeline



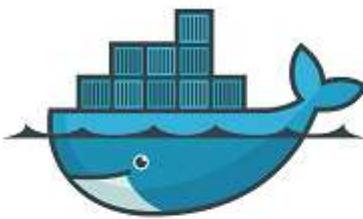
DevOps Pipeline on Google Cloud Platform



# 각 업체별 전략 – Google GCP Cloud Run

- 컨테이너를 프로덕션으로 신속히 배포 (Stateless, Serverless)
- 어디서든 동일한 환경 이용 가능 (On-Prem, GCP, Anthos)
- 100% 오픈소스 기반

Docker  
Container



Kubernetes



Cloud Run



Knative



Knative



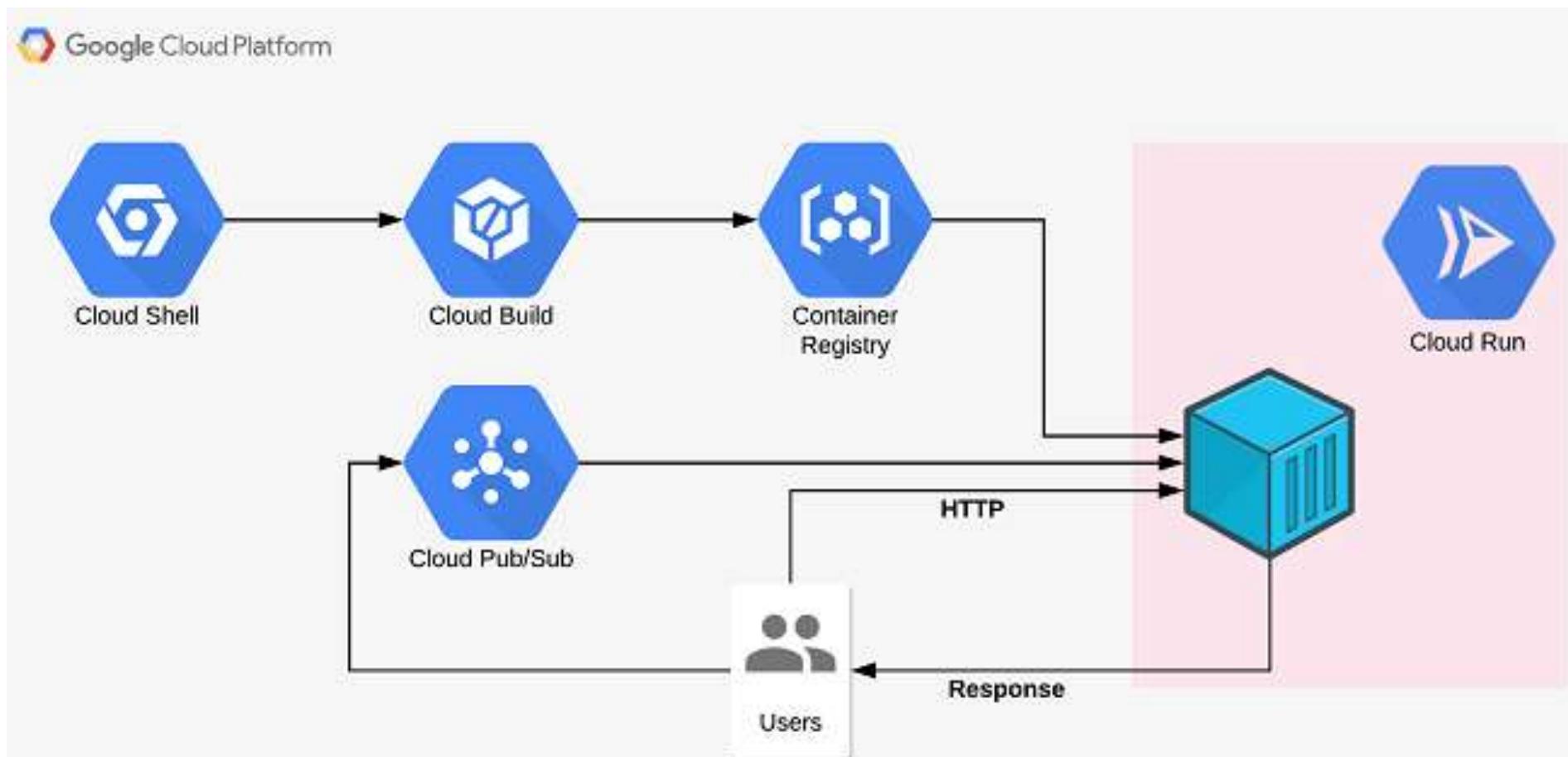
GKE  
(GCP Kubernetes  
Engine)

<https://cloud.google.com/run/?hl=ko>

<https://knative.dev/>

<https://medium.com/google-cloud/knative-to-cloud-run-f0ed1617e256>

# 각 업체별 전략 – Google GCP Cloud Run

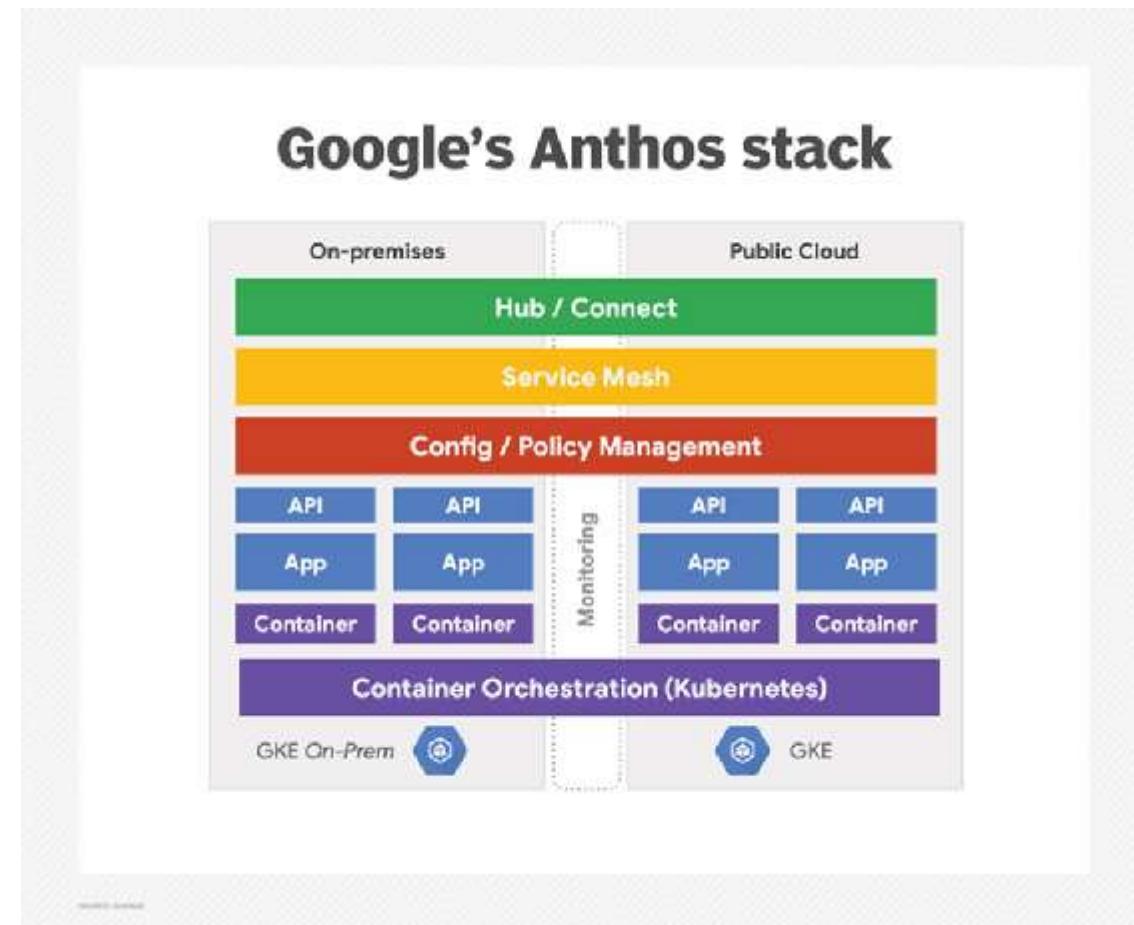
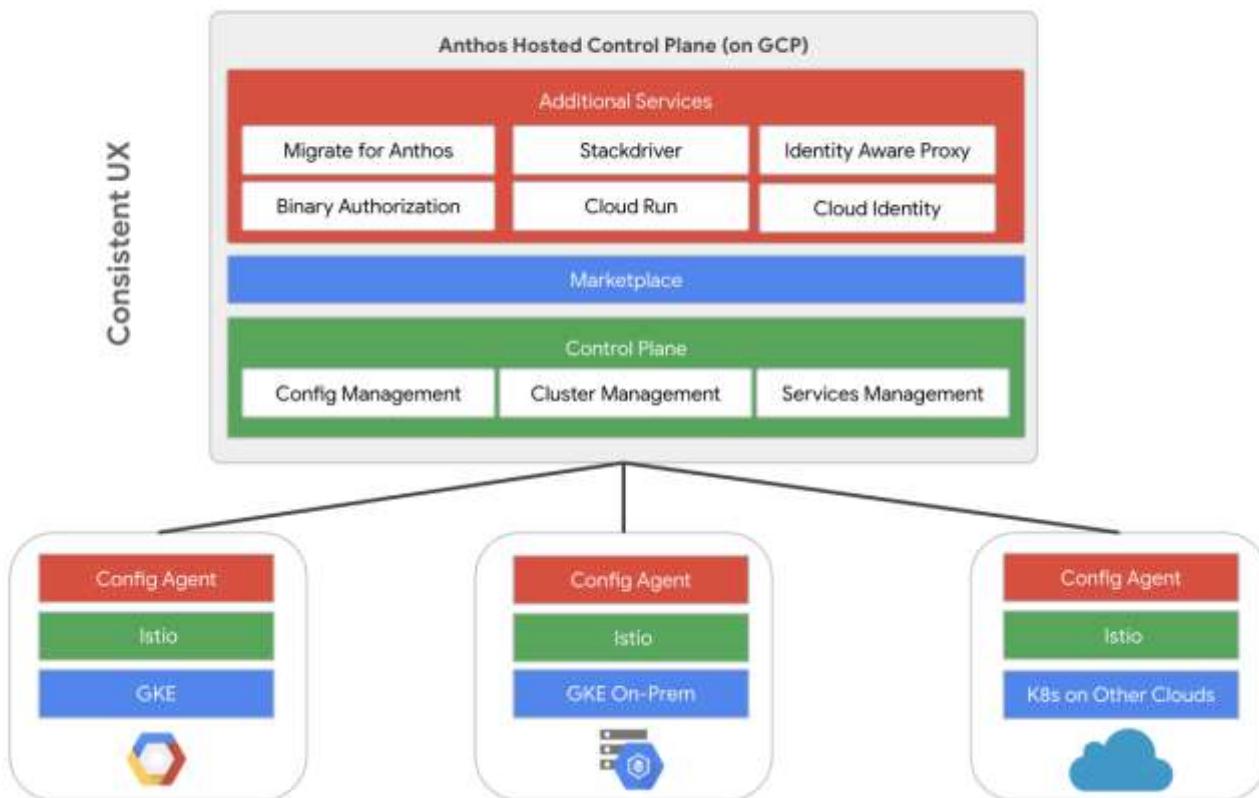


<https://towardsdatascience.com/cloud-run-dataset-summaries-via-http-request-9b5fe24fe9c1>

# 각 업체별 전략 – Google GCP Anthos



## Anthos: Hybrid Ecosystem



<https://codelabs.developers.google.com/codelabs/anthos-workshop/#0>

## Module 4

# Amazon EKS를 이용한 Kubernetes 배포운영 이해와 실습

# **Part I**

## **AWS 소개**

# 퍼블릭 클라우드 업체 비교: Gartner Magic Quadrant

2014년 7월



2017년 7월



2019년 7월



# AWS 장점

 <b>축적된 경험</b>	2006년부터 14년의 클라우드 서비스 운영 경험
 <b>서비스 분야 및 전문성</b>	다양한 클라우드 업무 지원하는 100여개 이상 서비스
 <b>혁신의 속도</b>	고객의 피드백에 기반한 빠른 혁신속도
 <b>글로벌 인프라</b>	22개 리전, 69개 가용 영역(AZ), 205개 엣지 로케이션
 <b>가격 철학</b>	자발적인 가격인하
 <b>파트너 생태계</b>	수많은 파트너사 및 7,300개 이상 마켓플레이스 제품

<https://aws.amazon.com/ko/about-aws/>

# AWS 차별점: 가장 넓고 많은 생태계



partner  
network

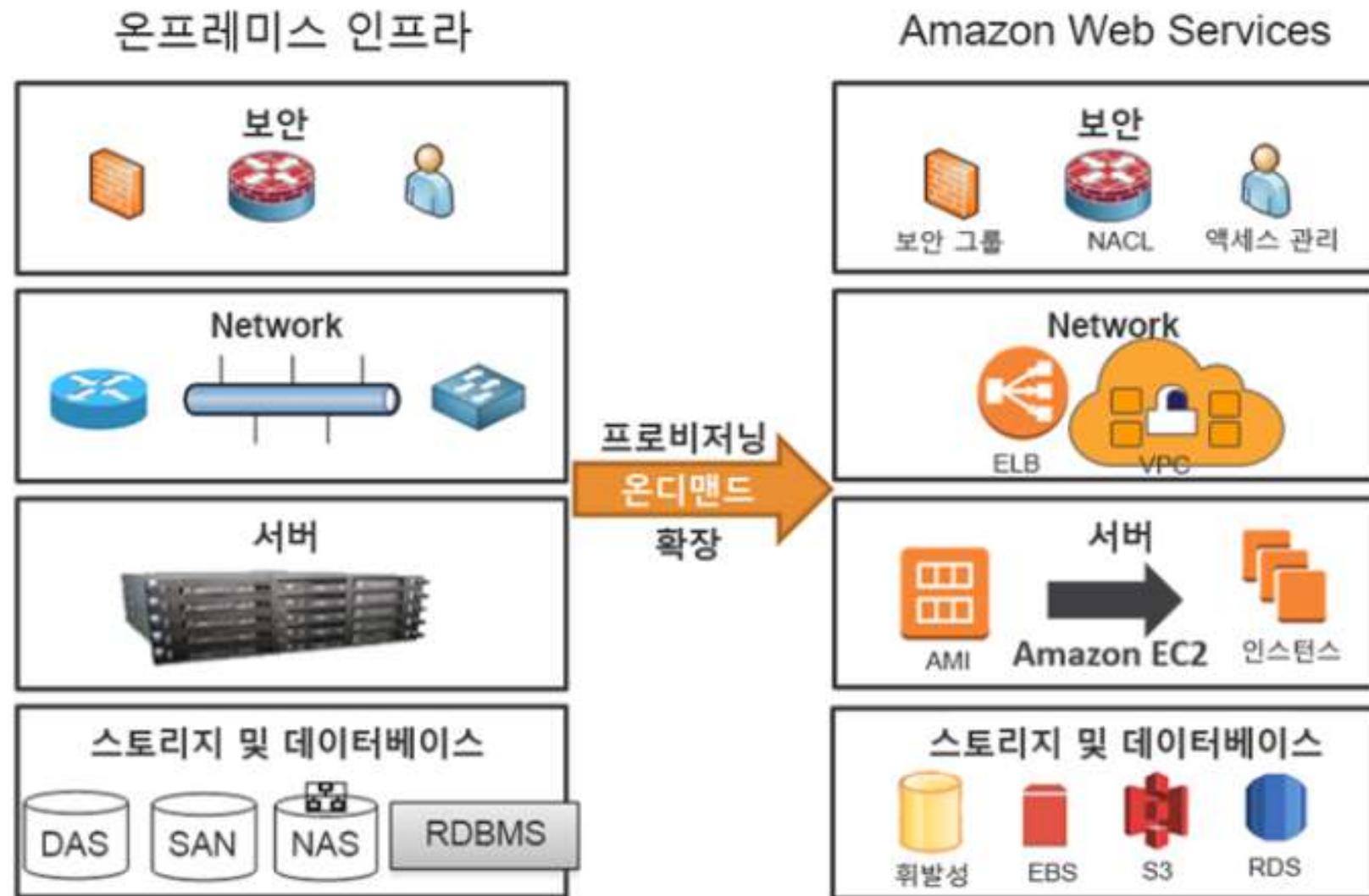
수만 개의 SI & Consultancy 및  
ISV 파트너



수십개의 제품 분류에 7,300개가 넘는  
파트너 제품이 있으며, 고객이 원클릭으  
로 설치 가능



# AWS 플랫폼



# AWS 클라우드 글로벌 인프라

AWS 클라우드는 전 세계 22개의 지리적 리전(Region) 내에 69개의 가용 영역(Availability Zone)을 운영. 서울 리전에는 3개의 가용영역이 있음 (2012년 사무소, 2016년 첫 리전 시작)

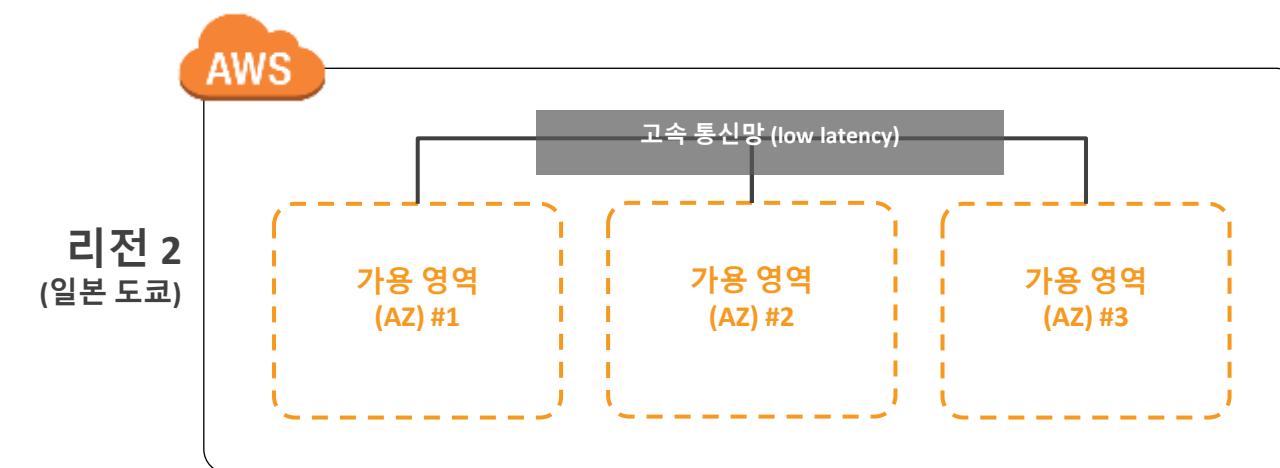
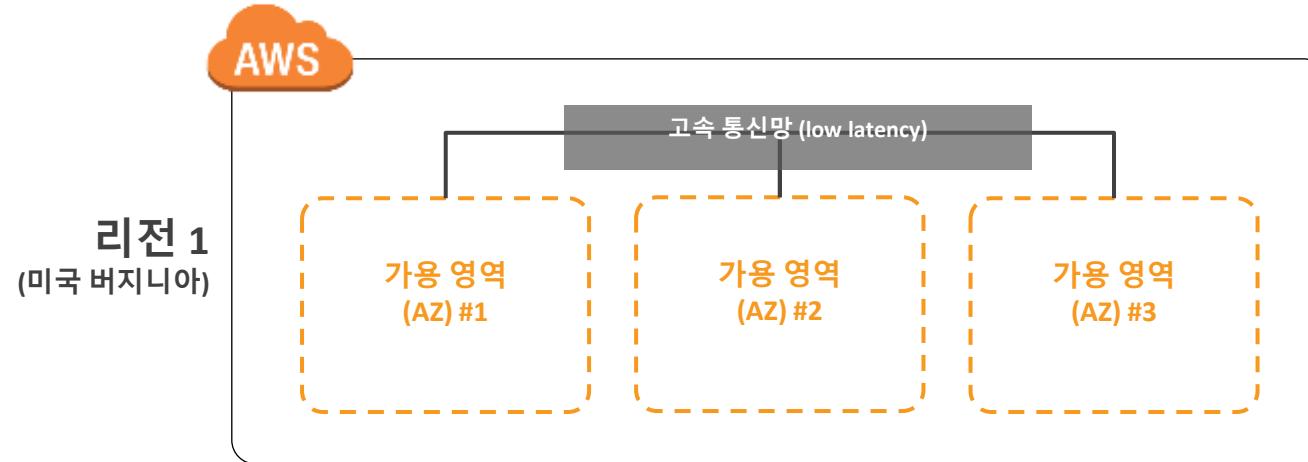


**22개 리전**  
(Region)

**69개 가용 영역**  
(Availability Zones)

**205개 엣지 로케이션**

# AWS 클라우드 글로벌 인프라 구성



- **리전** 은 AWS 서비스가 운영되는 지역으로 복수개의 데이터 센터들의 집합

- **가용 영역 (AZ)** 는 리전 내 위치한 복수 개의 데이터 센터들로 각각 물리적으로 분리되어 있어 고가용성/이중화 구성의 기본 요소를 형성

- **엣지 로케이션** 은 CloudFront 같은 엣지 서비스의 캐시 서버 (POP)가 운영되는 데이터센터

# AWS 리전 (Regions)

- AWS 리전은 두 개 이상의 가용 영역으로 이루어진 지리 영역입니다.
- 리전 선택은 다음 요소에 영향을 미칩니다.
  - 자연 시간 최적화
  - 비용 최적화
  - 규제 요건
- 완전히 독립적인 엔티티
- 퍼블릭 인터넷을 통해 리전 간 통신 발생
  - 전송 시 데이터 암호화



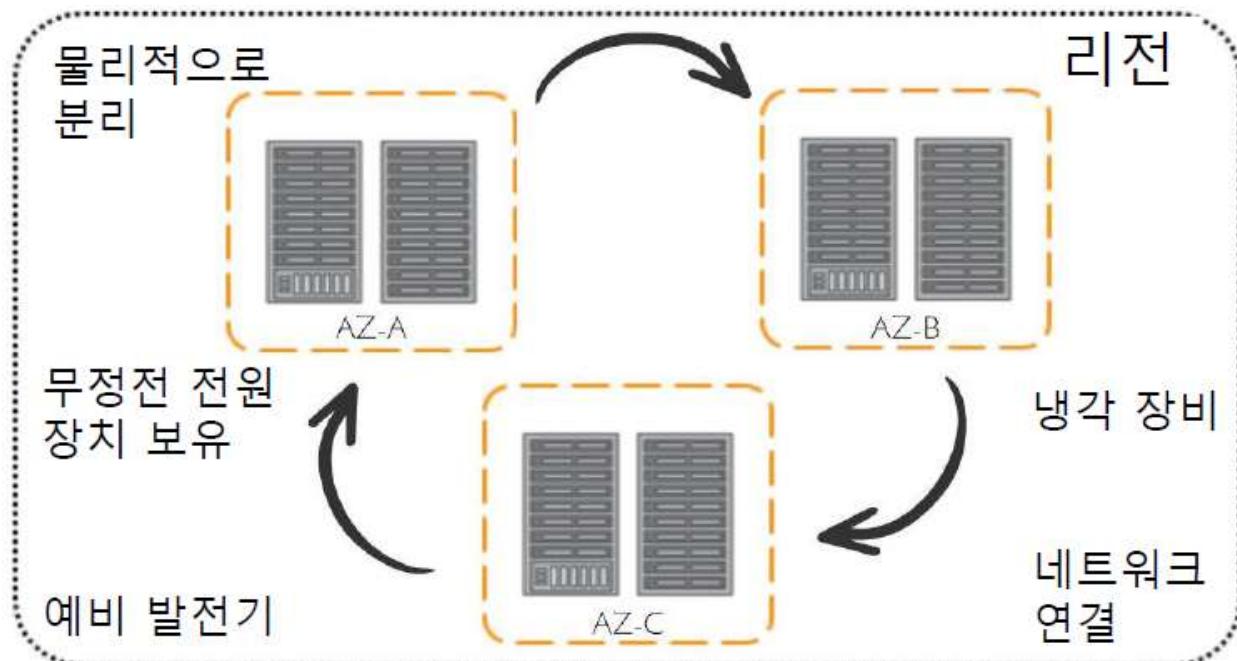
# AWS 가용 영역 (Availability Zones)

- 가용 영역은 각 리전 내에 존재하는 데이터 센터 모음임
- 각 가용 영역은 격리됨
- 지연 시간이 짧은 빠른 네트워크 링크로 연결
- AZ의 리소스를 장애로부터 보호
- 장애 발생 시 요청 처리
- 모범 사례: 다중 AZ에 리소스 프로비저닝

## 가용 영역 격리

- 장애로부터 영역을 보호
- 고가용성을 제공하도록 설계됨
- 다른 영역을 통해 요청을 처리

모범 사례: 다중 가용 영역을 구현



# 엣지 위치 (Edge Location)

- Amazon CloudFront
- Amazon Route 53
- AWS Shield
- AWS Web Application Firewall
- Lambda@Edge 컴퓨팅



## **Part II**

# **AWS 서비스와 주요자원 소개**

# AWS 서비스 종류



컴퓨팅



스토리지 및  
콘텐츠 전송



데이터베이스



네트워킹



관리 도구



보안 및  
자격 증명



분석



애플리케이션  
서비스



엔터프라이즈  
애플리케이션



인공 지능



IoT



AWS 개발자  
도구

# AWS 서비스 포트폴리오



# 컴퓨팅 서비스

The screenshot shows the AWS Management Console interface. At the top, there is a dark header bar with the AWS logo on the left, followed by the text "서비스 ▾" and "리소스 그룹 ▾". On the far right of the header is a small star icon. Below the header, on the left, is a sidebar with two items: "내역" (History) and "콘솔 홈" (Console Home). The main content area has a search bar at the top with the placeholder text "이름 또는 기능(예: EC2, S3 또는 VM, ...)". To the right of the search bar is a large, light blue rectangular area. On the left side of this area, there is a vertical list of services under the heading "컴퓨팅" (Computing), which includes EC2, Lightsail, ECR, ECS, EKS, Lambda, Batch, Elastic Beanstalk, Serverless Application Repository, AWS Outposts, and EC2 Image Builder. To the right of the service list, the text "EC2: Elastic" is displayed.

내역

콘솔 홈

이름 또는 기능(예: EC2, S3 또는 VM, ...)

컴퓨팅

- EC2
- Lightsail ↗
- ECR
- ECS
- EKS
- Lambda
- Batch
- Elastic Beanstalk
- Serverless Application Repository
- AWS Outposts
- EC2 Image Builder

EC2: Elastic

# 스토리지 서비스

The screenshot shows the AWS Storage Services console. At the top, there's a dark header bar with the AWS logo, a '서비스' dropdown menu, a '리소스 그룹' dropdown menu, and a star icon. On the left, a sidebar has '내역' and '콘솔 홈' buttons. The main area has a search bar with placeholder text '이름 또는 기능(예: EC2, S3...)' and a list of storage services:

- 스토리지** (with a folder icon)
- S3
- EFS
- FSx
- S3 Glacier
- Storage Gateway
- AWS Backup

# EBS & ELB

The screenshot shows the AWS EC2 console interface. At the top, there's a banner for the 'New EC2 Experience' with the message: '① 새로운 EC2 콘솔을 시작합니다. AWS는 사용 편의성을 높이고 성능을 개선하기 위해 EC2 콘솔을 재설계하고 있습니다. 주기적으로 새 화면을 필리스 할 예정입니다. 새로운 화면을 사용해 보는 데 편리하려면 [New EC2 Experience] 토글을 사용하십시오.' Below the banner, the main navigation menu on the left includes 'EC2 대시보드 New', '이벤트', '태그', '보고서', '제한', '인스턴스' (selected), '인스턴스 유형', '시작 템플릿 New', '스팟 요청', 'Savings Plans', '예약 인스턴스', '전용 호스트', '용량 예약', 'AMI', '변동 작업', and 'ELASTIC BLOCK STORE' (highlighted with a red box). The main content area displays '리소스' (Resources) for the Asia Pacific (Seoul) region, showing 0 items for each category: 실행 중인 인스턴스, 전용 호스트, 볼륨, 키 페어, 배치 그룹, 탄력적 IP, 스냅샷, 로드 밸런서, and 보안 그룹. A callout box provides information about deploying Microsoft SQL Server Always On availability groups. At the bottom, sections for '인스턴스 시작' (Instance Start) and '서비스 상태' (Service Status) are shown, along with a status summary: 리전 - 아시아 태평양 (서울), 상태 - 이 서비스가 정상적으로 작동 중입니다.

① 새로운 EC2 콘솔을 시작합니다.  
AWS는 사용 편의성을 높이고 성능을 개선하기 위해 EC2 콘솔을 재설계하고 있습니다. 주기적으로 새 화면을 필리스 할 예정입니다. 새로운 화면을 사용해 보는 데 편리하려면 [New EC2 Experience] 토글을 사용하십시오.

EC2

## 리소스

아시아 태평양 (서울) 리전에서 다음 Amazon EC2 리소스를 사용하고 있음:

실행 중인 인스턴스	0	탄력적 IP	0
전용 호스트	0	스냅샷	0
볼륨	0	로드 밸런서	0
키 페어	0	보안 그룹	1
배치 그룹	0		

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. [Learn more](#)

### ELASTIC BLOCK STORE

볼륨  
스냅샷  
수명 주기 관리자

### 인스턴스 시작

Amazon EC2 사용을 시작하려면 Amazon EC2 인스턴스라고 하는 가상 서버를 시작해야 합니다.

서비스 상태

리전  
아시아 태평양 (서울)

상태  
이 서비스가 정상적으로 작동 중입니다.

# 데이터베이스 서비스

The screenshot shows the AWS Management Console interface. At the top, there is a dark header bar with the AWS logo on the left, followed by the text "서비스 ▾" and "리소스 그룹 ▾". On the far right of the header is a small star-shaped icon. Below the header, on the left side, is a sidebar with two items: "내역" (History) and "콘솔 홈" (Console Home). The main content area has a search bar at the top with the placeholder text "이름 또는 기능(예: EC2, S3 또는 VM)". Below the search bar, there is a section titled "데이터베이스" (Database) with a database icon. This section lists several services: RDS, DynamoDB, ElastiCache, Neptune, Amazon Redshift, Amazon QLDB, Amazon DocumentDB, and Managed Cassandra Service.

내역

콘솔 홈

이름 또는 기능(예: EC2, S3 또는 VM)

데이터베이스

- RDS
- DynamoDB
- ElastiCache
- Neptune
- Amazon Redshift
- Amazon QLDB
- Amazon DocumentDB
- Managed Cassandra Service

# 네트워크/VPC 서비스

The screenshot shows the AWS Management Console with the following interface elements:

- Header:** AWS logo, Service dropdown, Resource Groups dropdown, and a star icon.
- Left Sidebar:** Navigation links for "내역" (History) and "콘솔 홈" (Console Home).
- Search Bar:** A search input field with placeholder text "이름 또는 기능(예: EC2, S3 또는".
- Service List:** A list of network and content delivery services, each with an icon:
  - VPC**: Cloud icon, labeled "네트워킹 및 콘텐츠 전송".
  - CloudFront**: Cloud icon.
  - Route 53**: Cloud icon.
  - API Gateway**: Cloud icon.
  - Direct Connect**: Cloud icon.
  - AWS App Mesh**: Cloud icon.
  - AWS Cloud Map**: Cloud icon.
  - Global Accelerator**: Cloud icon.

VPC : Virtual Private Cloud

# 관리/보안 서비스



## 관리 및 거버넌스

AWS Organizations  
CloudWatch  
AWS Auto Scaling  
CloudFormation  
CloudTrail  
Config  
OpsWorks  
Service Catalog  
Systems Manager  
AWS AppConfig  
Trusted Advisor  
Control Tower  
AWS License Manager  
AWS Well-Architected Tool  
Personal Health Dashboard ↗  
AWS Chatbot  
Launch Wizard

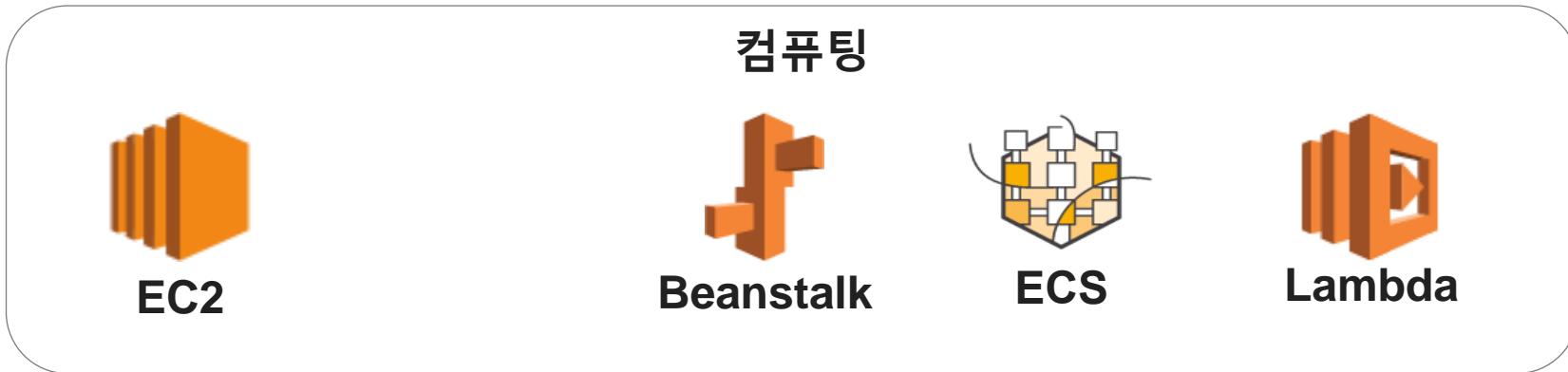


## 보안, 자격 증명 및 규정 준수

IAM  
Resource Access Manager  
Cognito  
Secrets Manager  
GuardDuty  
Inspector  
Amazon Macie ↗  
AWS Single Sign-On  
Certificate Manager  
Key Management Service  
CloudHSM  
Directory Service  
WAF & Shield  
Artifact  
Security Hub  
Detective

# AWS 컴퓨팅 서비스

# 컴퓨팅 서비스



- AWS
  - 유연성
  - 비용 효율적
- Amazon EC2
  - 유연한 구성 및 제어
- AWS Lambda
  - 사용한 만큼만 비용을 지불
  - 관리 불필요
- Amazon Lightsail
  - 가상 프라이빗 서버 시작
  - 간편한 웹 및 애플리케이션 서버 관리
- Amazon ECS
  - 관리형 컨테이너
  - 뛰어난 확장성, 뛰어난 성능
- AWS Fargate
- Amazon EKS

# 컴퓨팅 서비스 소개



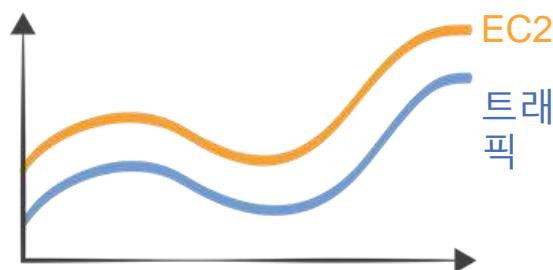
“가상 서버 서비스”



Auto Scaling



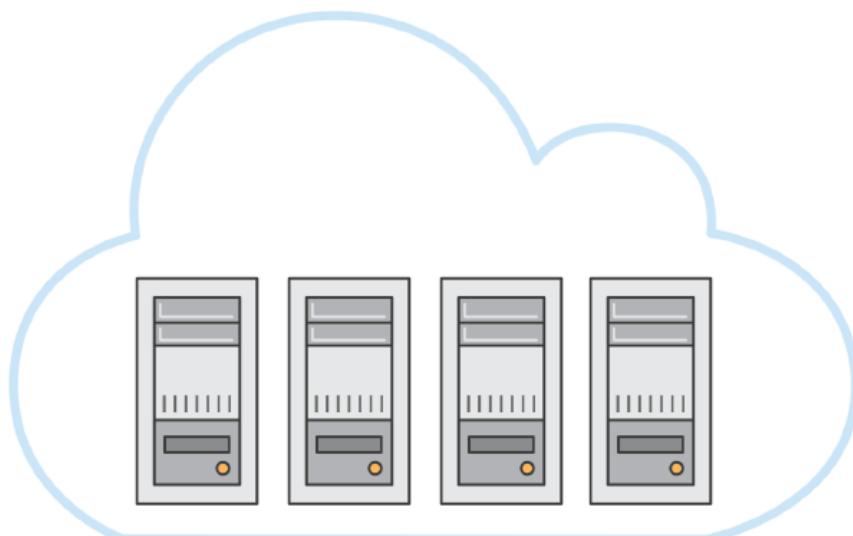
“서버 자동 확장/축소”



“서비스 컴퓨팅”



## 탄력적인 컴퓨팅 클라우드



- 애플리케이션 서버
- 웹 서버
- 데이터베이스 서버
- 게임 서버
- 메일 서버
- 미디어 서버
- 카탈로그 서버
- 파일 서버
- 컴퓨팅 서버
- 프록시 서버

# Amazon EC2: 가상 서버 서비스



**Amazon EC2**  
(Elastic Compute Cloud)

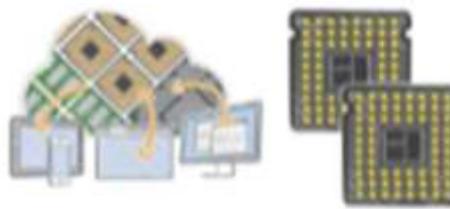
“가상 서버 서비스”



- Virtual Machine
- 재구성이 가능한 컴퓨팅 리소스
- 쉽게 확장/축소되는 컴퓨팅 용량
- ‘고객 업무’ 영역에 따른 다양한 인스턴스 타입 제공
- 사용한 만큼만 과금 (pay-as-you-go)

# Amazon EC2: 인스턴스 패밀리 범주

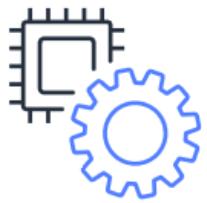
<https://aws.amazon.com/ko/ec2/>



## 범용

비즈니스 크리티컬 애플리케이션, 중소 규모 데이터베이스, 웹 티어 애플리케이션 등에 이상적입니다.

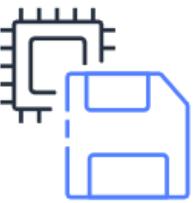
- M, T Family
- 컴퓨팅, 메모리, 네트워크 리소스의 균형적 사용
- 성능 순간 확장 가능 인스턴스 (T)



## 컴퓨팅 최적화

고성능 컴퓨팅, 일괄 처리, 비디오 인코딩 등에 이상적입니다.

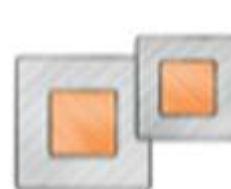
- C Family
- EC2에서 최고 성능의 프로세서, 성능 대비 저렴한 가격



## 메모리 최적화

고성능 데이터베이스, 분산 웹 규모 인 메모리 캐시, 실시간 빅 데이터 분석 등에 이상적입니다.

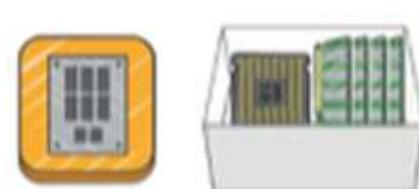
- R Family
- 메모리 용량이 많이 필요한 애플리케이션용



## 가속화된 컴퓨팅

기계 학습, 그래픽 집약적 애플리케이션, 게임 등에 이상적입니다.

- G, P Family
- 그래픽 및 일반 목적의 GPU 컴퓨팅 애플리케이션



## 스토리지 최적화

NoSQL 데이터베이스, 데이터 웨어하우징, 분산 파일 시스템 등에 이상적입니다.

- D, I Family
- 고밀도 높은 디스크 처리량, 단위당 최소의 가격 (D)
- SSD 기반의 높은 IO (I)

## Amazon EC2: 인스턴스 유형 및 사용사례

패밀리	설명	사용 사례 예시
t2, m3, m4, m5	범용 균형 잡힌 성능	웹 사이트, 웹 애플리케이션, 개발, 코드 리포지토리, 마이크로 서비스, 비즈니스 앱
c3, c4, c5, cc2	컴퓨팅 최적화 뛰어난 CPU 성능	프런트 엔드 플릿, 웹 서버, 배치 처리, 분산 분석, 과학 및 엔지니어링 앱, 광고 제공, MMO 게임, 비디오 인코딩
g2, p2	GPU 최적화 고성능 GPU	Amazon AppStream 2.0, 비디오 인코딩, 기계 학습, 고성능 데이터베이스, 과학
r3, r4, r5, x1, cr1	메모리 최적화 대규모 RAM 공간	인 메모리 데이터베이스, 데이터 마이닝
d2, i2, i3, hi1, hs1	스토리지 최적화 높은 I/O, 고밀도	NAS, 데이터 웨어하우징, NoSQL

# m5.large

- 
- The diagram illustrates the structure of an Amazon EC2 instance name. At the top center is the name "m5.large". A horizontal orange line extends from the right side of ".large" to the right. Three dashed orange lines descend from the left side of "m5.", each ending in a small horizontal bar. The first dashed line ends at the word "인스턴스". The second dashed line ends at "용도". The third dashed line ends at "인스턴스". To the right of "large", there is a single solid orange line extending downwards.
- 인스턴스 패밀리
  - 용도 별로 선택
    - 인스턴스 세대
    - 높을 수록 최신
    - 최신 = 비용 대비 성능이 우수
  - 인스턴스 사이즈
  - 커질 때마다 용량 및 가격이 ~2배씩 증가

# Amazon EC2: 인스턴스 선택법

- EC2 인스턴스 유형은 다양한 사용 사례 및 워크로드에 최적화 되었고 여러 크기로 제공. 이를 통해 워크로드 요구 사항에 따라 리소스를 최적으로 조정할 수 있음.
- AWS 에서는 EC2 인스턴스에 대해 인텔 제온 프로세서를 활용
- 인스턴스를 선택할 때 고려 사항
  - 코어 수
  - 메모리 크기
  - 스토리지 크기 및 유형
  - 네트워크 성능
  - I/O 요구 사항
  - CPU 기술
- 빨리 처리하고 쉬기 (HUGI)  
컴퓨팅 인스턴스가 클수록 시간과 비용을 절약할 수 있음. 즉, 더 짧은 시간 동안 시간당 비용이 높은 인스턴스를 실행하는 것이 더 경제적일 수 있음



# Amazon EC2 요금제

- **온디맨드**

온디맨드 인스턴스에서는 실행하는 인스턴스에 따라 시간당 또는 초당 컴퓨팅 파워에 대한 비용을 지불합니다. 장기 약정이나 선결제 금액은 필요 없습니다. 애플리케이션 수요에 따라 컴퓨팅 파워를 늘리거나 줄일 수 있으며 사용한 인스턴스에 대해 지정된 시간당 요금만 지불하면 됩니다.

- **스팟 인스턴스**

Amazon EC2 스팟 인스턴스를 사용하면 온디맨드 요금보다 최대 90% 할인된 가격으로 예비 Amazon EC2 컴퓨팅 용량을 요청할 수 있습니다.

- **Savings Plans**

Savings Plans는 1년 또는 3년 기간의 일정 사용량 약정(시간당 요금을 기준으로 측정)을 조건으로 EC2 및 Fargate 사용량에 대해 저렴한 요금을 제공하는 유연한 요금 모델입니다.

- **예약 인스턴스**

예약 인스턴스는 온디맨드 인스턴스 요금과 비교하여 상당한 할인 혜택(최대 75%)을 제공합니다. 또한, 예약 인스턴스를 특정 가용 영역에 지정하면 용량 예약이 제공되므로 필요할 때 예약한 인스턴스를 시작할 수 있다는 확신을 가질 수 있습니다.

- **전용 호스팅**

전용 호스팅은 고객 전용의 물리적 EC2 서버입니다. 전용 호스팅을 사용하면 Windows Server, SQL Server, SUSE Linux Enterprise Server(라이선스 약관에 따름)를 비롯한 기존 서버 한정 소프트웨어 라이선스를 사용할 수 있으므로 비용을 절감할 뿐 아니라 규정 준수 요구 사항도 충족할 수 있습니다.

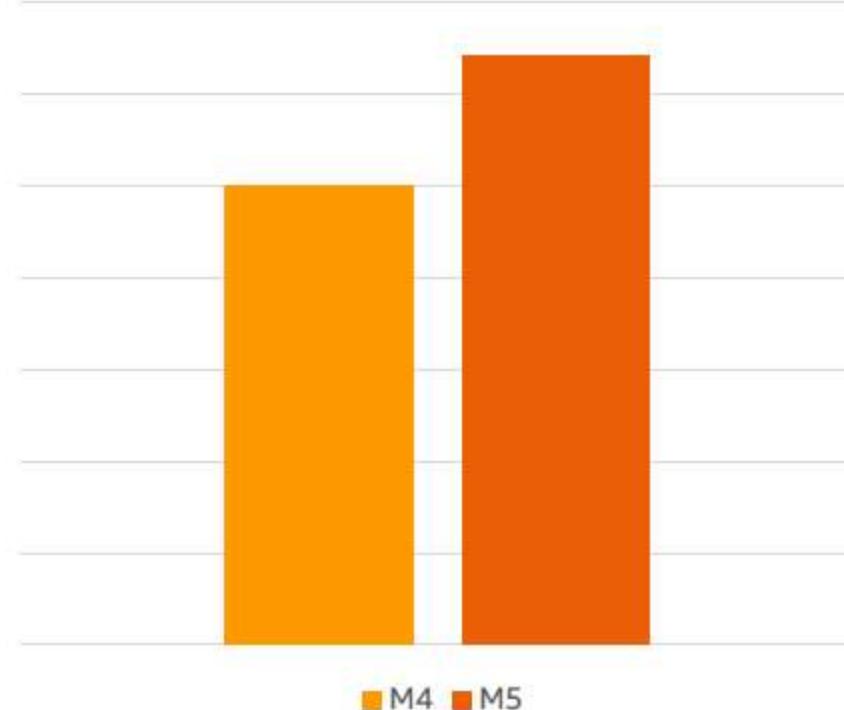
# 인텔 계열의 EC2 인스턴스

EC2 인스턴스 유형	컴퓨팅 최적화		범용			메모리 최적화			스토리지 최적화		
	C5n	C5	M5	T3	T2	X1	X1e	R4	H1	I3	D2
인텔 프로세서	제온 플래티넘 8175M	제온 플래티넘 8175M	제온 플래티넘 8175M	제온 플래티넘 8175M	제온 패밀리	제온 E7 8880 v3	제온 E7 8880 v3	제온 E5 2686 v4	제온 E5 2686 v4	제온 E5 2686 v4	제온 E5 2676 v3
인텔 프로세서 기술	Skylake	Skylake	Skylake	Skylake	예	Haswell	Haswell	Broadwell	Broadwell	Broadwell	Haswell
인텔 AVX	예	예	예	예	예	예	예	예	예	예	예
인텔 AVX2	예	예	예	예	-	예	예	예	예	예	예
인텔 AVX-512	예	예	예	예	-	-	-	-	-	-	-
인텔 터보 부스트	예	예	예	예	예	예	예	예	예	예	예
스토리지	EBS 전용	EBS 전용	EBS 전용	EBS 전용	EBS 전용	SSD EBS 옵션	SSD EBS 옵션	-	HDD	SSD	HDD

## M5: 차세대 범용 인스턴스

- 2.5 GHz 인텔 제온 확장형 프로세스(**Skylake**) 탑재
- 더 큰 크기의 새로운 인스턴스 - m5.24xlarge  
최대 vCPU 96개 및 메모리 384GiB (4:1 메모리 대 vCPU 비율)
- 더 작은 크기의 인스턴스보다 향상된 네트워크 및 EBS 성능
- 벡터 및 부동 소수점 워크로드에 대해 최대 2배의 성능을 제공하는 인텔 **AVX-512** 지원

M5에서 성능  
가격비 14% 향상



### T3: 차세대 범용 인스턴스

- 컴퓨팅, 메모리, 네트워크 리소스의 균형
- 필요한 경우 언제든지 기준선 수준의 CPU 성능에 CPU 사용량을 순간 확장할 수 있는 기능 제공
- 시간당 \$0.0052로 가장 낮은 비용의 인스턴스로, 인텔 제온 확장형 프로세서를 사용하여 T2 대비 최대 30% 개선된 성능 대비 가격 제공

t3.nano
0.5 GiB
2 vCPU
Base perf 5%

7 sizes  
● ● ●

t3.2xlarge
32 GiB
8 vCPU
Base perf 40%

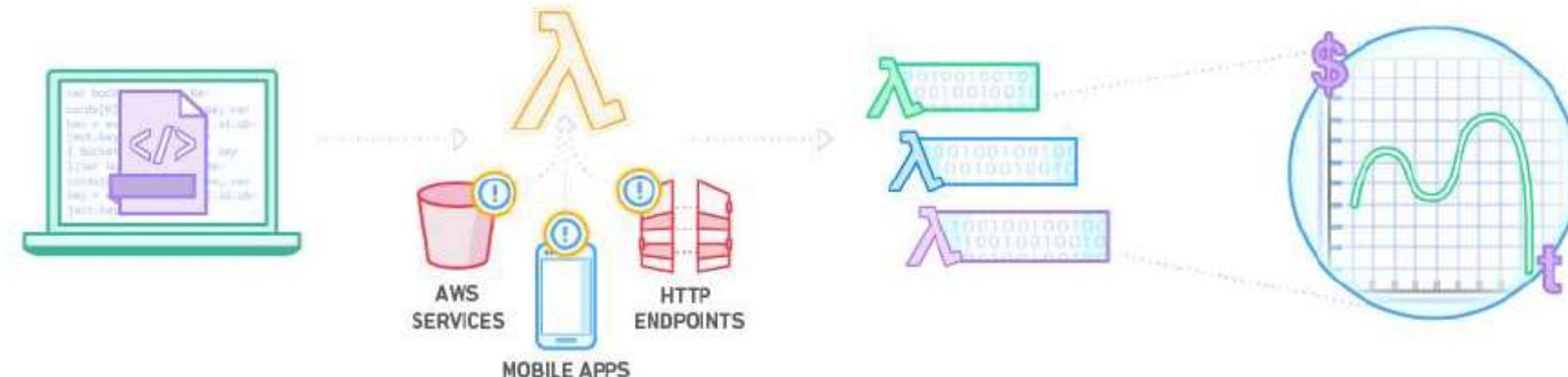


T3는 기본 성능을 초과하는 무제한 버스팅이 24시간 평균, vCPU-시간당 \$0.05 요금



# AWS Lambda: 서버리스 컴퓨팅

“서버 없이 코드만으로 특정 업무를 처리 – 이벤트 처리 기반”



Node.js  
Python  
Java  
C#

예) S3에 이미지 업로드 시

예) 해당 이미지 리사이징

ms 시간 단위 과금

# EC2 실습

- AWS 콘솔에 로그인합니다.
- EC2 마법사를 시작합니다.
- AMI(SW)를 선택합니다.
- 인스턴스 유형을 선택합니다.
- 네트워크를 구성합니다.
- 스토리지를 구성합니다.
- 프라이빗 키를 수집합니다.
- 시작합니다.
- 연결합니다.

# AWS 스토리지 서비스

# 스토리지 및 컨텐츠 배포 서비스 소개



**Amazon S3**  
(Simple Storage Service)

“객체 스토리지”



이미지  
비디오  
파일  
스냅샷 등

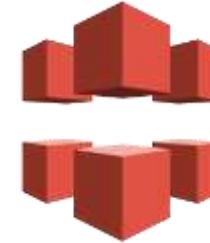


**Amazon EBS**  
(Elastic Block Store)

“블록 스토리지”

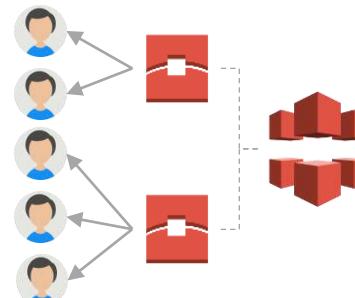


EC2에 attach 해서 사용



**Amazon CloudFront**

“컨텐츠 전송 네트워크”



캐싱을 통한  
컨텐츠 가속

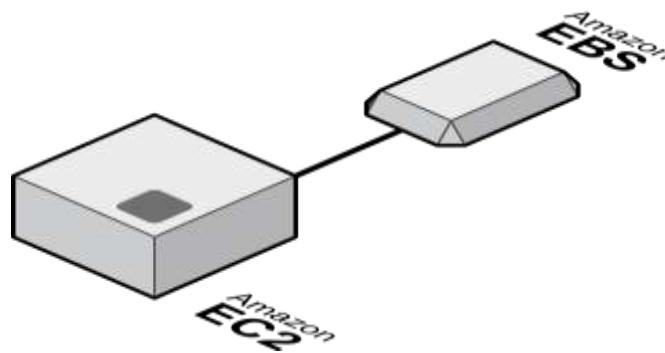
# Amazon EBS: 블록 스토리지

- **EBS: Elastic Block Store**
- EC2 인스턴스를 위한 사용자 지정 가능한 영구 블록 스토리지
- HDD 및 SSD 유형
- 백업용 스냅샷
  - 특정 시점 스냅샷
  - 언제든 새로운 볼륨을 다시 생성
- 쉽고 투명한 암호화
  - 암호화된 EBS 볼륨
  - 추가 비용 없음
- 탄력성
  - 용량 증가
  - 다른 유형으로 변경
- 가용성: 내구성이 있으며 자동으로 동일한 가용 영역에 복제됨

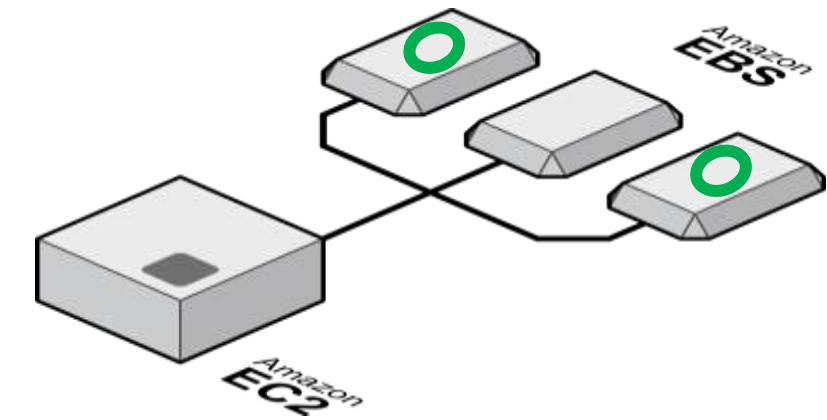
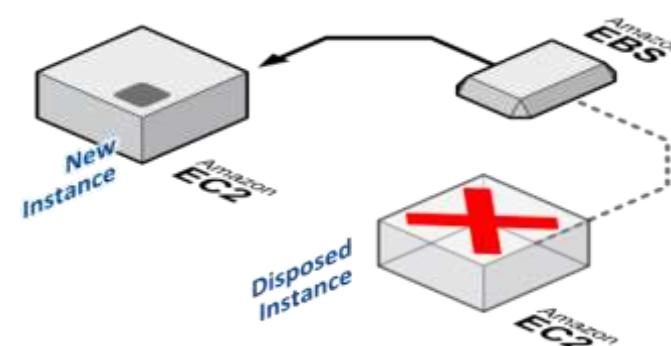
# Amazon EBS: 블록 스토리지

## EC2에 attach 해서 쓸 수 있는 블록 스토리지

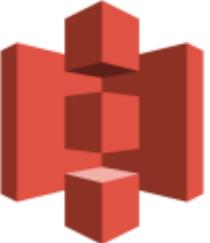
- ✓ 단일 가용 영역 내에서 여러 서버에 걸쳐 복제
- ✓ 특정 시점에 대한 볼륨 스냅샷을 만들 수 있는 기능 제공 → S3에 저장되어 복수의 가용 영역 (AZ)에 걸쳐 자동으로 복제



한 개의 EBS가 여러 개의 EC2에 attach 불가  
한 개의 EC2에 여러 개의 EBS를 attach 가능



# Amazon S3: 무제한 객체 스토리지



**Amazon S3**  
(Simple Storage Service)

“객체 스토리지”



이미지  
비디오  
파일  
스냅샷 등

- 객체 기반의 **무제한 파일 저장 스토리지**
- **URL을 통해 손쉽게 파일 공유 가능**
- **99.999999999%의 내구성**
- **사용한 만큼만 지불 (GB 당 과금)**
- **정적 웹 사이트 호스팅 가능**

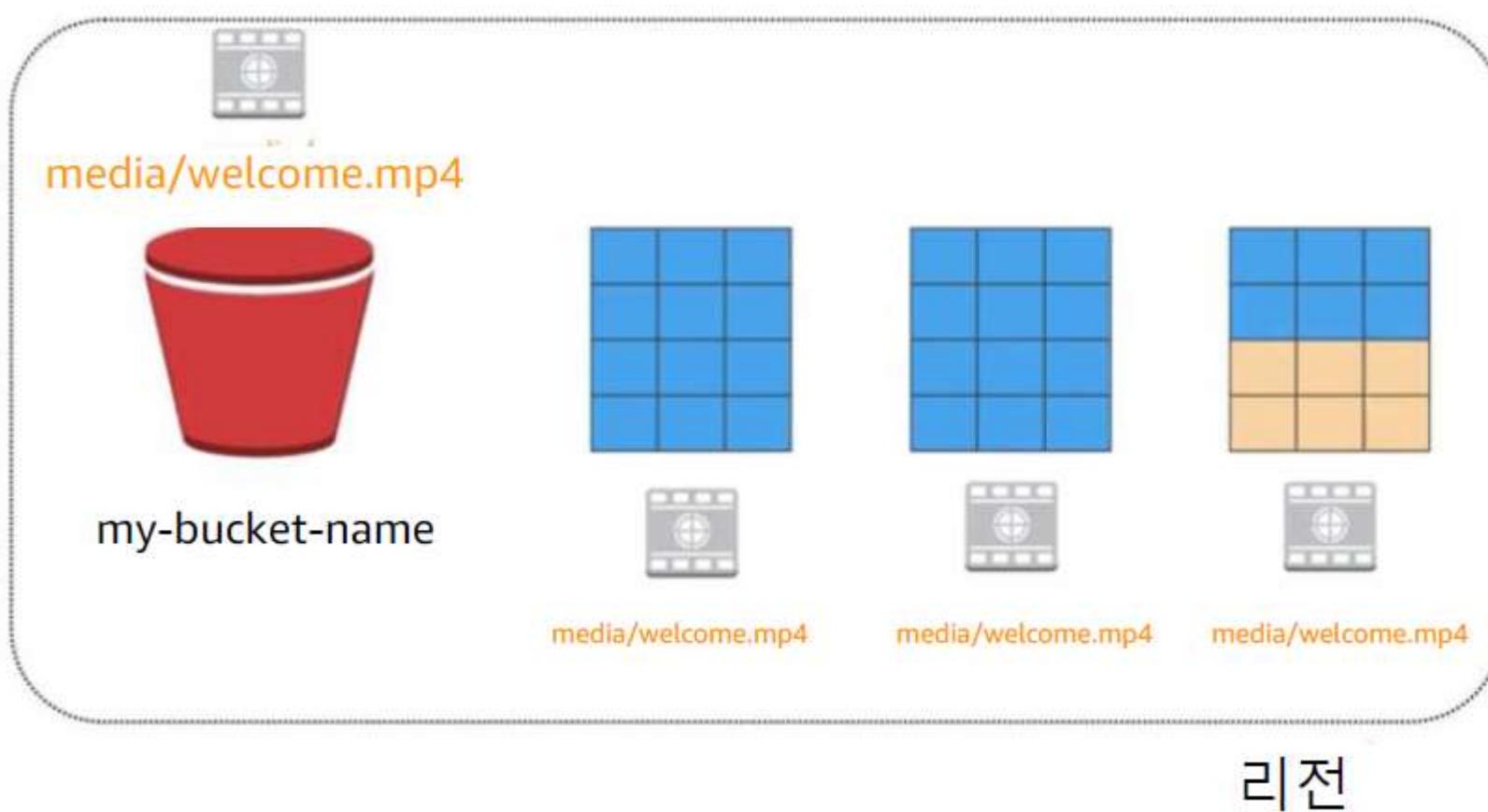
## Amazon S3: 개요

- Simple Storage Solution
- 가장 기본적인 storage service, 데이터의 전체 볼륨과 객체 수에는 제한 없음
- 각 객체에 최대 5TB 데이터 저장 가능하며, 고유한 키를 사용하여 저장 및 검색
- 사용자별 액세스 권한 부여 가능 및 개별 버킷 정책 연결 가능
- 표준 기반 REST 인터페이스와 AWS SDK 지원
- 저장 되는 기본은 객체이며, 객체 데이터와 메타데이터로 구성(객체를 설명하는 이름-값 페어의 집합, 콘텐츠 형식, 수정한 날짜 등)
- 객체는 키(이름, 단일하고 고유) 및 버전 ID를 통해 버킷 내에서 고유하게 식별  
예) <http://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl>이라는 URL에서 "doc"는 버킷의 이름이고 "2006-03-01/AmazonS3.wsdl"은 키
- Amazon S3는 모든 리전의 덮어쓰기 PUTS 및 DELETES에 대한 최종 일관성을 제공
- 처음부터 모든 인터넷 애플리케이션의 트래픽을 처리할 수 있도록 설계
- 해당 버킷은 해당 AWS 계정의 소유, 계정당 기본 최대 100개, 추가 신청시 1000개까지

# Amazon S3: Naming

- 버킷 이름은 Amazon S3에 있는 어떤 기존 버킷 이름과도 중복되지 않아야 함
- 해당 버킷은 해당 AWS 계정의 소유
- 계정당 기본 최대 100개, 추가 신청시 1000개까지 가능
- 버킷을 만든 후에는 버킷 리전을 변경할 수 없음
- DNS 이름 지정 규칙을 준수해야 하며, IP 주소 형식(예: 192.168.5.4)을 사용하지 않음
- 3자 이상, 63자 이하
- 대문자나 밑줄을 사용할 수 없으며(하이픈"-" 가능), 소문자나 숫자로 시작하고 끝나야 함
- 버킷 이름에 마침표(".")를 사용하지 않는 것이 좋음
- 각 버킷을 <https://myawsbucket.s3.amazonaws.com>과 같이 가상 호스트 스타일 주소 지정을 사용하여 주소 지정 가능

# Amazon S3: 리전에 데이터 중복 저장

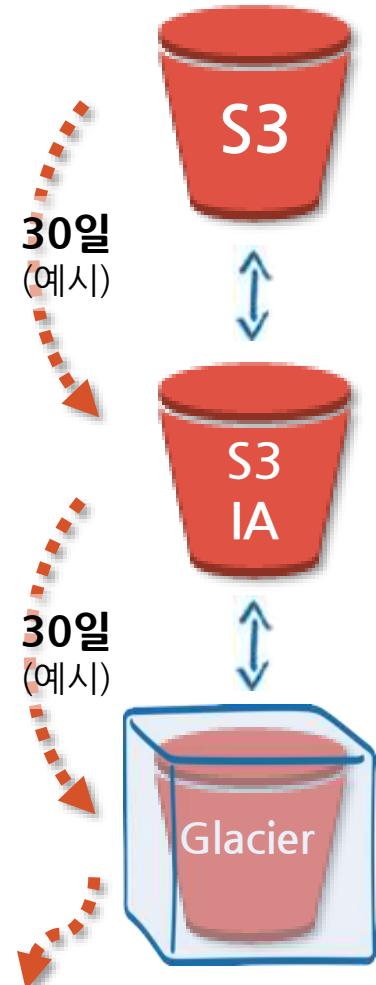


# Amazon S3: AWS 서비스들과의 통합



“다양한 AWS 서비스들과의 통합/연계 지원”

# Amazon S3: 스토리지 옵션 및 Glacier, 라이프사이클 관리



## Amazon S3 – 무제한 스토리지

- ✓ 99.999999999% 내구성 / 99.99% 가용성
- ✓ 최대 저장 가능 객체 크기 5TB / 수는 무제한

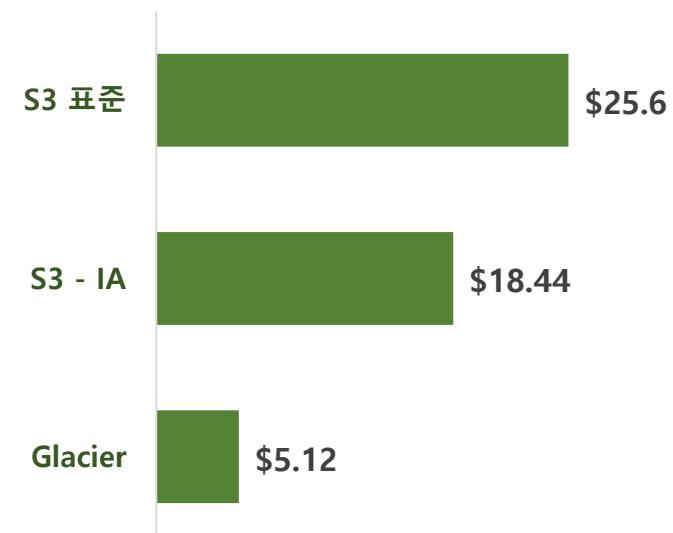
## Amazon S3 – Infrequent Access Storage

- ✓ S3와 같은 내구성 및 성능 / 99.9% 가용성
- ✓ 자주 접근하지 않는 데이터를 저렴하게 보관
- ✓ 자주 조회하는 데이터에는 부적합 (조회 비용 때문)

## Amazon Glacier – 데이터 백업

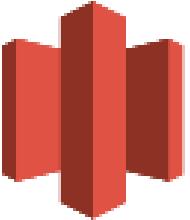
- ✓ S3와 같은 내구성, 성능 및 가용성 / 3가지 데이터 검색 옵션
- ✓ 아카이빙, 장기간 백업 및 오래된 로그 데이터

“상황에 맞는  
비용 효율적인 저장을 고려하자!”  
- 1TB 저장에 소요되는 비용



\* 2017년 9월 서울 Region Simple Monthly Calculator 기준  
<http://calculator.s3.amazonaws.com/index.html>

# Amazon Glacier: 아카이빙/백업 스토리지



## Amazon Glacier

### “아카이빙 백업 스토리지”



이미지  
비디오  
파일  
스냅샷 등

- 99.999999999%의 내구성
- 백업 / 아카이빙 용도의 Cold 데이터
- 사용한 만큼 매우 낮은 비용으로 활용 가능

#### Amazon S3 Glacier Deep Archive(S3 Glacier Deep Archive)

##### 데이터 검색

스탠다드 GB당 0.022 USD

대량 GB당 0.005 USD

##### 데이터 검색 요청

스탠다드 요청 1,000건당 0.1086 USD

대량 요청 1,000건당 0.0275 USD

S3 Glacier Deep Archive에 대한 PUT 요청

요청 1,000건당 0.06 USD

S3 Glacier Deep Archive로 수명 주기 전환 요청

요청 1,000건당 0.06 USD

# S3 사용 모범사례

- 동일 리전에 S3와 인스턴스 설치 (네트워크 지연시간 감소, 데이터 전송비용 절감)
- 올바른 정책 사용
  - S3 퍼블릭 액세스 차단
  - “\*”와 같은 와일드카드 자격 증명이나 작업을 허용하는 버킷 정책을 금지
  - “인증된 모든 AWS 사용자”에게 읽기, 쓰기 권한을 제공하는 ACL(액세스 제어 목록)을 금지
  - 관리형 AWS Config 규칙 사용(s3-bucket-public-read-prohibited & s3-bucket-public-write-prohibited)
- 최소 권한 액세스 구현 (IAM & ACL) 및 애플리케이션 및 AWS 제품에 IAM 역할 임시 자격 증명 사용
- Multi-Factor Authentication(MFA) 삭제 활성화로 실수로 삭제할 가능성 방지
- 전송 중 데이터의 암호화 적용 (HTTPS(TLS)를 사용)
- 버전 관리 사용 및 유휴 데이터의 암호화
- S3 액세스의 VPC 엔드포인트 고려
- Tag 사용하여 S3 식별 및 감사
- S3 서버 액세스 로깅 활성화 및 CloudTrail 사용

# S3 실습

aws 서비스 리소스 그룹 ★

Amazon S3 버킷 검색 모든 액세스 유형

버킷 만들기 버클리 액세스 설정 편집 비우기 삭제 0 버킷 0 리전

버킷이 없습니다. 다음은 Amazon S3를 시작하는 방법입니다.

 버킷 새로 만들기

버킷은 Amazon S3에 저장하는 모든 항목에 대한 전역적으로 고유한 컨테이너입니다.

 데이터 업로드

버킷을 만든 후 객체(예: 사진 또는 동영상 파일)를 업로드할 수 있습니다.

 권한 설정

기본적으로 객체에 대한 권한은 프라이빗이지만, 액세스 제어 정책을 설정하여 다른 사용자에게 권한을 부여할 수 있습니다.

세부 정보 세부 정보 세부 정보

시작하기

# AWS 데이터베이스 서비스

# 데이터베이스 서비스 소개



**Amazon RDS**  
(Relational Database Service)

“관리형 관계형  
DB 서비스”

자동 백업  
이중화

Aurora, Oracle  
MSSQL, PostgreSQL  
MySQL, MariaDB,



**Amazon DynamoDB**

“관리형 NoSQL  
DB 서비스”

높은 확장성

10ms 미만의 지연시간



**Amazon ElastiCache**

“인메모리 캐싱 서비스”

Memcached  
Redis

# Amazon RDS

- 클라우드에서 관계형 데이터베이스를 설정 및 운영하는 관리형 서비스
- <https://aws.amazon.com/ko/rds/>



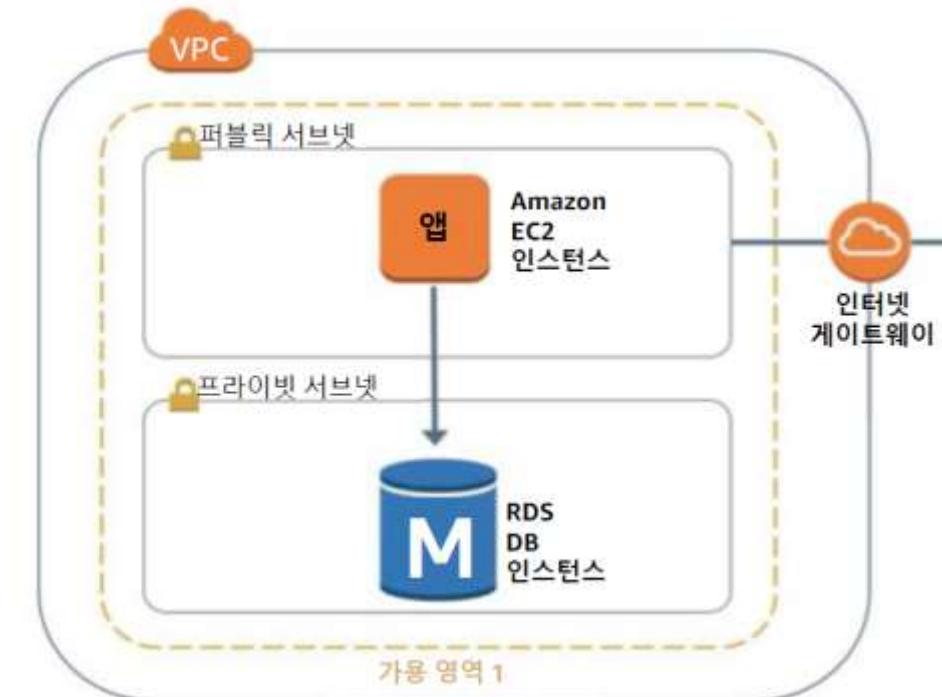
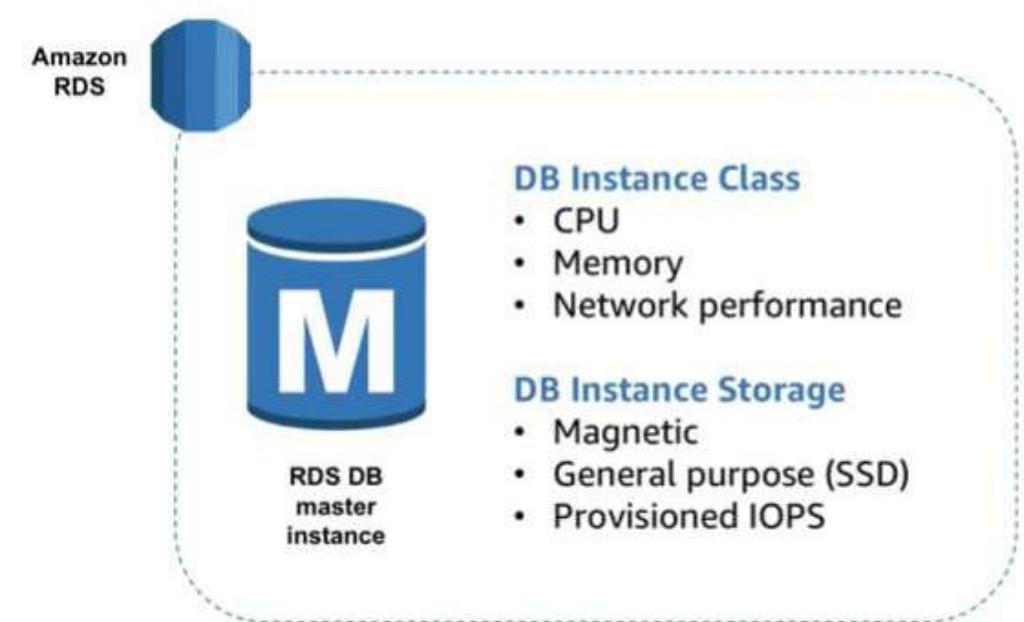
## 고객 관리 항목

- 애플리케이션 최적화

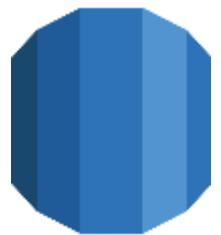
## AWS 관리 항목

- OS 설치 및 패치
- 데이터베이스 소프트웨어 설치 및 패치
- 데이터베이스 백업
- 고가용성
- 조정
- 전력, 랙 및 스택
- 서버 유지 관리

# Amazon RDS DB 인스턴스



# Amazon RDS: Managed 관계형 데이터베이스 서비스

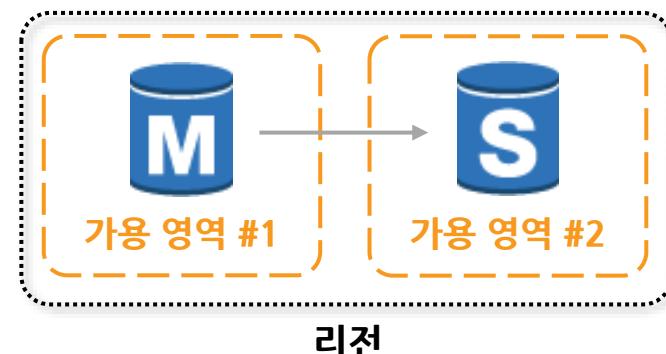


**Amazon RDS**

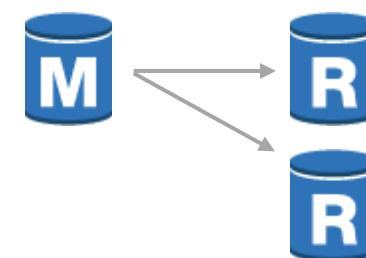
(Relational Database Service)

“완전 관리형 관계형  
DB 서비스”

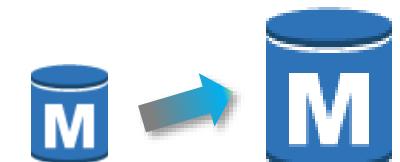
## DB 이중화 (Multi-AZ)



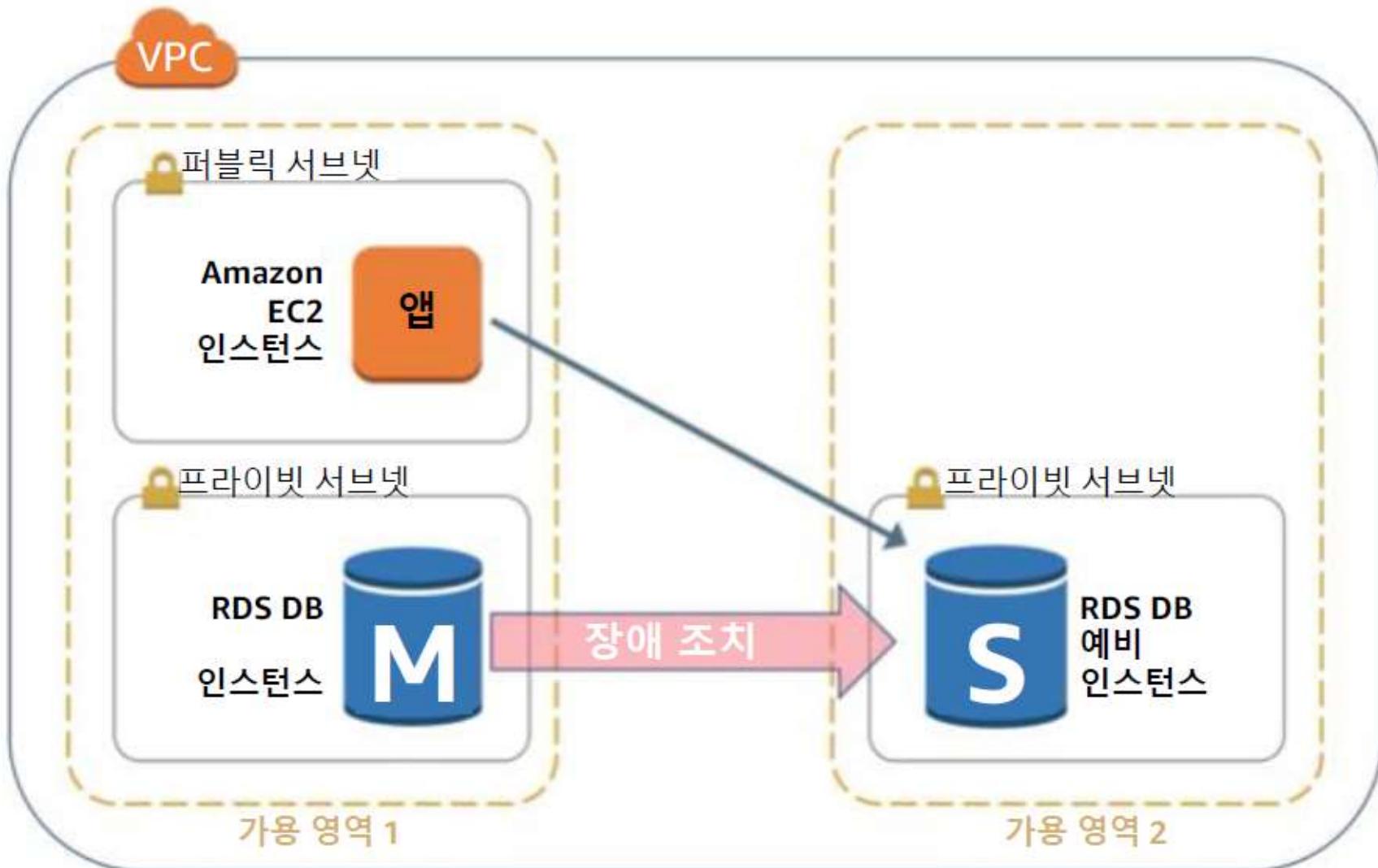
## Read Replica



## 인스턴스 확장

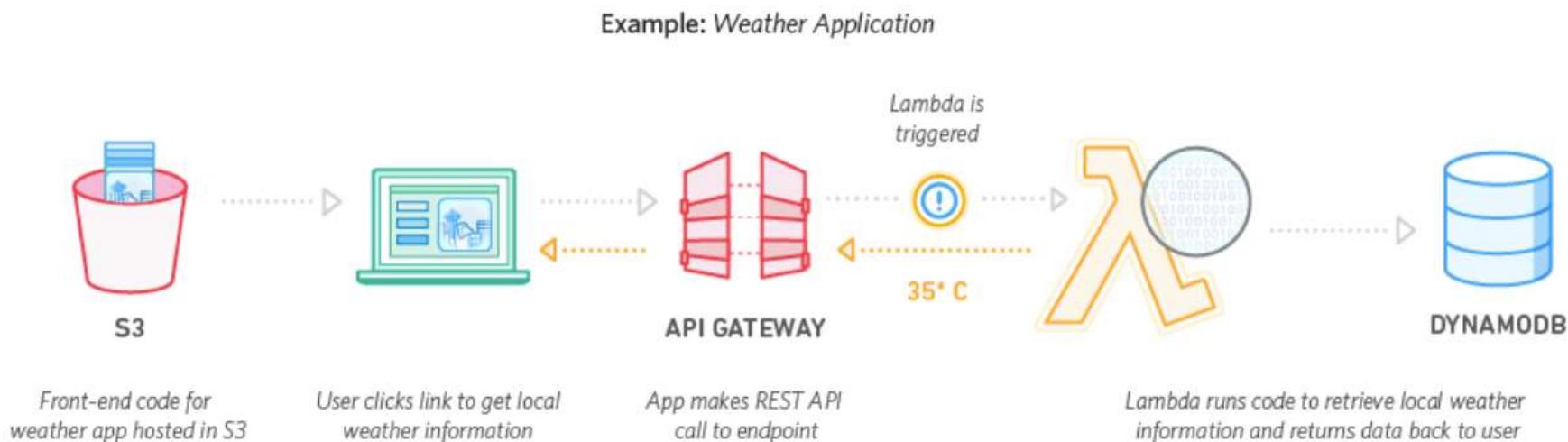


# Amazon RDS: 다중 AZ 를 통한 고가용성



# Amazon DynamoDB

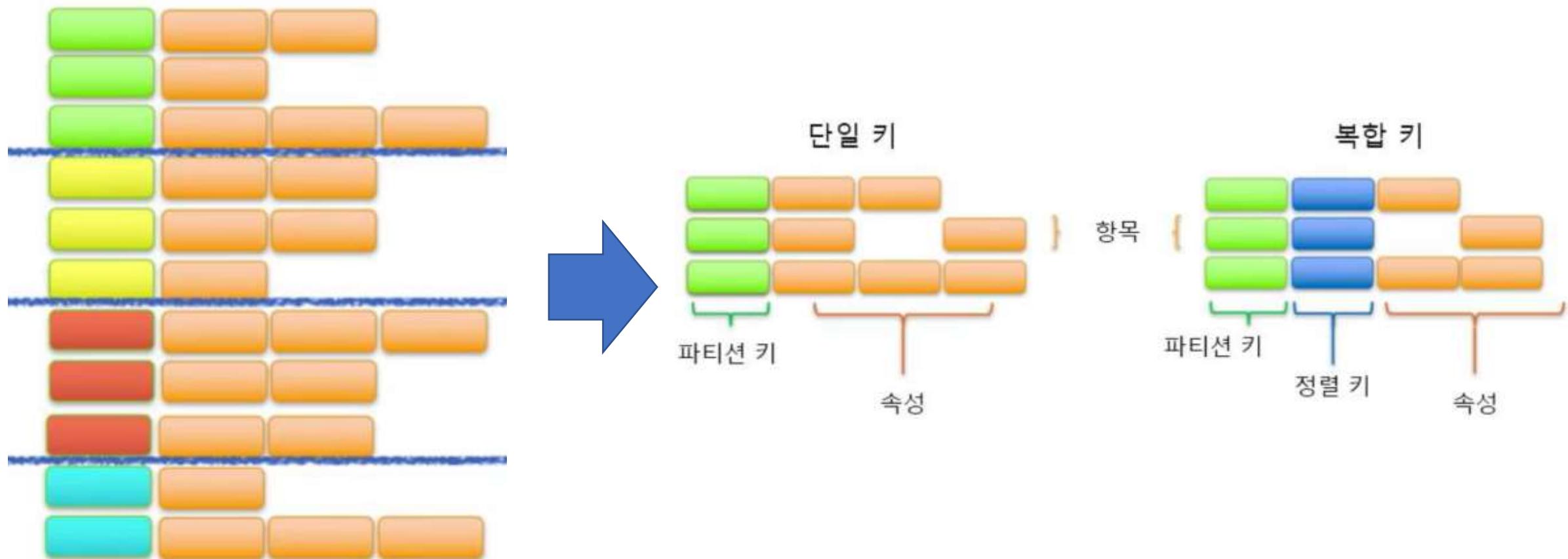
- 완전관리형의 내구성이 뛰어난 다중 리전, 다중 마스터 NoSQL 키-값 및 문서 데이터베이스
- 어떤 규모에서도 10밀리초 미만의 성능을 제공
- <https://aws.amazon.com/ko/dynamodb/>



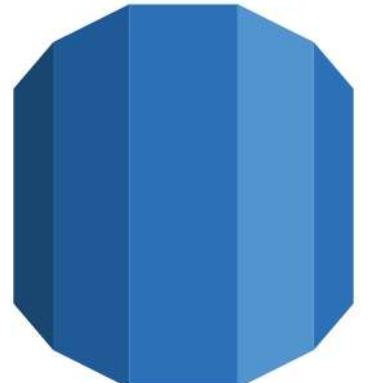
- NoSQL 데이터베이스 테이블
- 사실상 무제한의 스토리지
- 서로 다른 속성을 가질 수 있는 항목
- 자연 시간이 짧은 쿼리
- 확장 가능한 읽기 쓰기 처리량

# Amazon DynamoDB: 파티셔닝 (Partitioning)

- 데이터가 증가함에 따라 키로 테이블(Table)을 파티셔닝
- 키(Key)로 쿼리(Query)하여 항목을 효율적으로 검색 스캔하여 속성별로 항목



# Amazon Aurora: 성능 및 비용효율성을 모두 확보한 DB 엔진



**Amazon Aurora**



## 호환성

MySQL 5.6 완벽 호환  
PostgreSQL 9.6 완벽호환



## 성능 향상

기존 MySQL 대비 5배  
PostgreSQL 대비 3배



고가용성 및  
내구성



상용 DB엔진 대비  
1/10 가격



가장 빠르게 성장하는  
AWS 서비스

(PostgreSQL 호환 버전 정식 출시 예정)

# 데이터베이스 실습

The screenshot shows the AWS Amazon RDS Dashboard. At the top, there's a banner about Amazon Aurora, followed by sections for resources, database creation, and key features.

**Amazon Aurora**  
Amazon Aurora는 MySQL 및 PostgreSQL 호환 엔터프라이즈급 데이터베이스입니다(시작 비용 <\$1/일). Aurora는 최대 64TB의 Auto Scaling 스토리지 용량, 3개의 사용 영역에 대한 6 방향 복제, 지연 시간이 짧은 15개의 읽기 전용 복제본을 지원합니다. [자세히 알아보기](#)

[데이터베이스 생성](#)

또는, [S3에서 Aurora DB 클러스터 복원](#)

**리소스**

Refresh

Category	Count
DB 인스턴스 (0/40)	할당된 스토리지 (0 바이트/100.00 TB)
예약 인스턴스 (0/40)	여기클릭하여 DB 인스턴스 한도 높이기
스냅샷 (19)	수동 (0/100) 자동화 (0) 최근 이벤트 (0) 이벤트 구독 (0/20)
파라미터 그룹 (0)	기본값 (0) 사용자 지정 (0/100) 옵션 그룹 (0) 기본값 (0) 사용자 지정 (0/20) 서브넷 그룹 (0/50)
기본 네트워크 vpc-c84e86a3	지원되는 플랫폼 VPC

**추가 정보**

- RDS 시작하기
- 개요 및 기능
- 설명서
- 도움말 및 자습서
- MySQL용 데이터 가져오기 설명서
- Oracle용 데이터 가져오기 설명서
- SQL Server용 데이터 가져오기 설명서
- 새로운 RDS 기능 발표
- 요금
- 포럼

**데이터베이스 생성**

Amazon Relational Database Service(RDS)는 클라우드에 데이터베이스를 쉽게 생성, 운영 및 확장할 수 있는 웹 서비스입니다.

[S3에서 복원](#) [데이터베이스 생성](#)

참고: DB 인스턴스가 Asia Pacific (Seoul) 리전에서 시작합니다

**주요 기능**

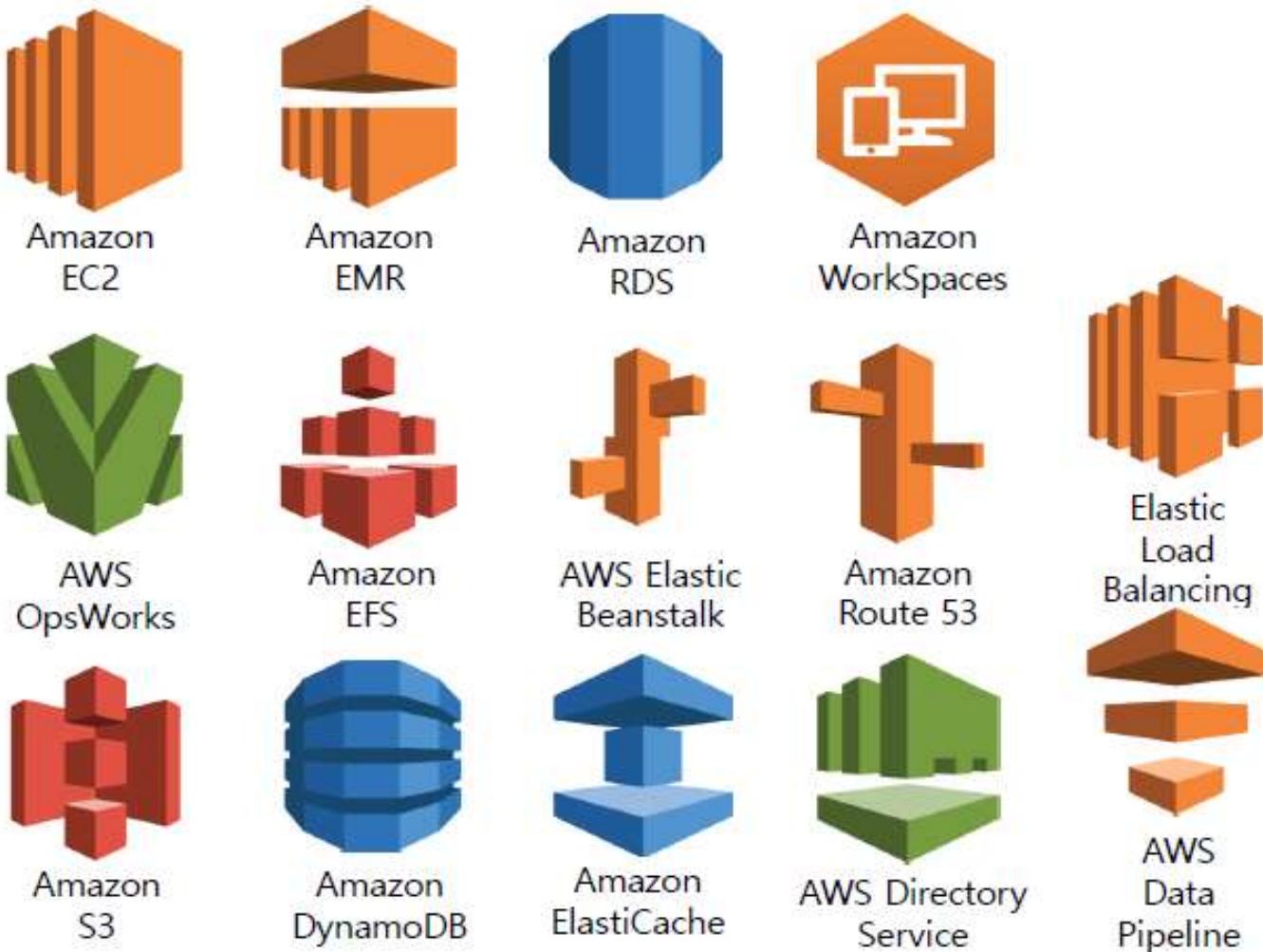
신규 기능: Elasticsearch를 위한 오픈 소스 Distro  
Elasticsearch의 100% 오픈 소스 및 커뮤니티 주도형 배포판으로서, 엔터프라이즈급 보안과 경고 기능을 제공합니다. [자세히 알아보기](#)

# **AWS Virtual Private Cloud (VPC)**

# Amazon Virtual Private Cloud (VPC)

- AWS VPC 소개
  - AWS 클라우드 내 프라이빗 가상 네트워크
  - 온프레미스 네트워크와 비슷한 구조
  - 네트워크 구성에 대한 완벽한 제어
- 배포
  - 보안 제어 계층 추가
  - AWS 서비스는 네트워크에 구축된 보안을 상속

# Amazon Virtual Private Cloud (VPC)



# Amazon Virtual Private Cloud (VPC)

## 기능

- 특징
  - 가상 네트워크를 프로비저닝하도록 허용
- 논리적으로 격리됨
- 구성 가능한 주요 기능
  - IP 범위
  - 라우팅
  - 네트워크 게이트웨이
  - 보안 설정
- 라우팅 테이블
  - 서브넷으로부터 전송되는 트래픽을 제어

# Amazon Virtual Private Cloud (VPC)

The screenshot shows the AWS VPC Dashboard. At the top, there's a navigation bar with the AWS logo, service dropdown, resource group dropdown, and account information (devops\_test, Seoul, Japan). Below the navigation is the main content area.

**VPC 대시보드** (VPC Dashboard) is the active tab. A sidebar on the left lists various VPC-related services: 가상 프라이빗 클라우드 (Virtual Private Cloud), VPC, 서브넷, 라우팅 테이블, 인터넷 게이트웨이, 외부 전용 인터넷 게이트웨이, DHCP 옵션 세트, 탄력적 IP, 엔드포인트, 엔드포인트 서비스, NAT 게이트웨이, and 피어링 연결. The main content area displays the following information:

- 리전별 리소스** (Regional Resources):
  - VPC: 1 item (모든 리전 보기)
  - 서브넷: 3 items (모든 리전 보기)
  - 라우팅 테이블: 1 item (모든 리전 보기)
  - 인터넷 게이트웨이: 1 item (모든 리전 보기)
  - 외부 전용 인터넷 게이트웨이: 0 items (모든 리전 보기)
- EC2 인스턴스 시작** (Launch EC2 Instances):

참고: 인스턴스는 아시아 태평양(서울) 리전에서 시작됩니다.
- 서비스 상태** (Service Status):

현재 상태	세부 정보
Amazon EC2 - 아시아 태평양(서울)	서비스가 정상적으로 작동 중입니다.
- 계정 속성** (Account Attributes):

리소스 ID 길이 관리
- 추가 정보** (Additional Information):

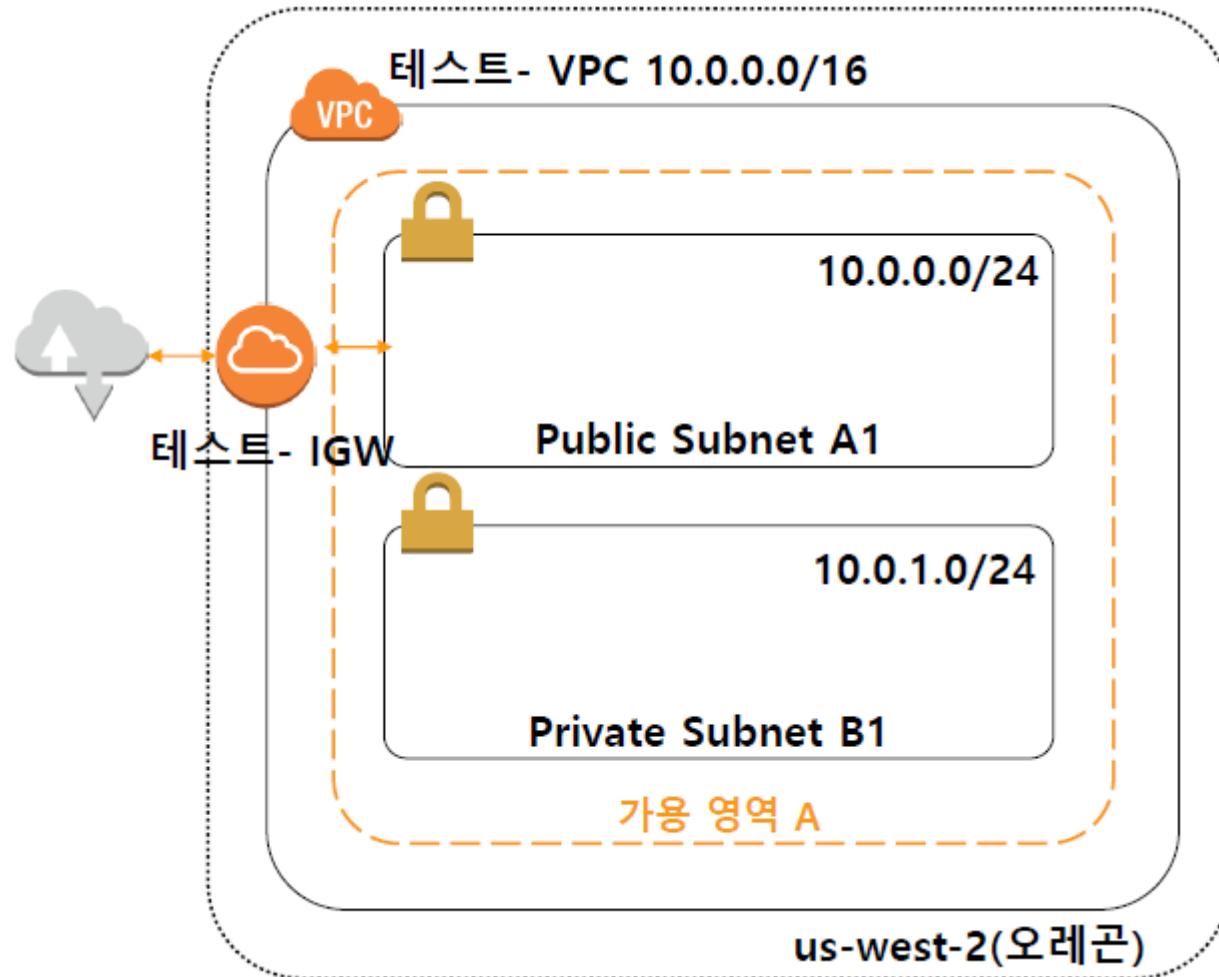
VPC 설명서, 모든 VPC 리소스, 포럼, 문제 보고
- Transit Gateway Network Manager**:

Network Manager enables centrally manage your global network across AWS and on-premises. [Learn more](#)

# Amazon Virtual Private Cloud (VPC)

## 실습

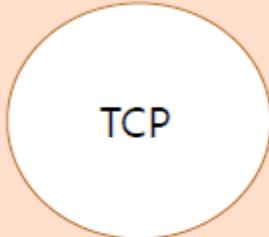
- 특정 리전에 VPC 생성
- 인터넷 게이트웨이 생성
- 퍼블릭 서브넷 생성
- 프라이빗 서브넷 생성



# **AWS Elastic Load Balancer (ELB)**

# AWS ELB

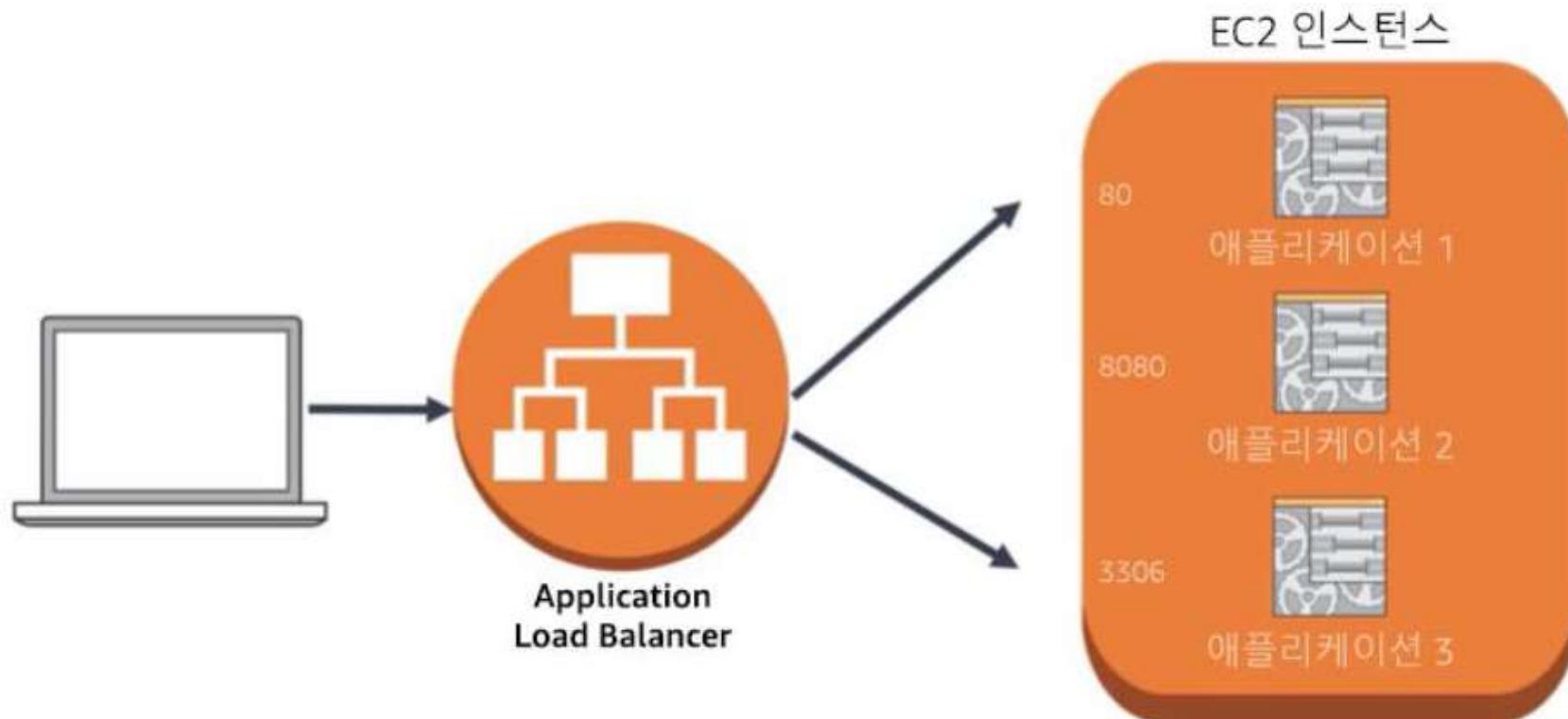
- 관리형 Load Balancing Service
- Instance 간에 부하를 분산

Application Load Balancer (ALB)	Network Load Balancer (NLB)	Classic Load Balancer (CLB)
		HTTP, HTTPS 및 TCP용 이전 세대
<ul style="list-style-type: none"><li>· 유연한 애플리케이션 관리</li><li>· HTTP 및 HTTPS 트래픽 고급 로드 밸런싱</li><li>· 요청 수준 (7계층)에서 작동</li></ul>	<ul style="list-style-type: none"><li>· 애플리케이션에 탁월한 성능 및 고정 IP 제공</li><li>· TCP 트래픽 로드 밸런싱</li><li>· 연결 수준 (4계층)에서 작동</li></ul>	<ul style="list-style-type: none"><li>· EC2-Classic 네트워크 내에서 구축된 기존 애플리케이션</li><li>· 요청 수준 및 연결 수준에서 모두 작동</li></ul>

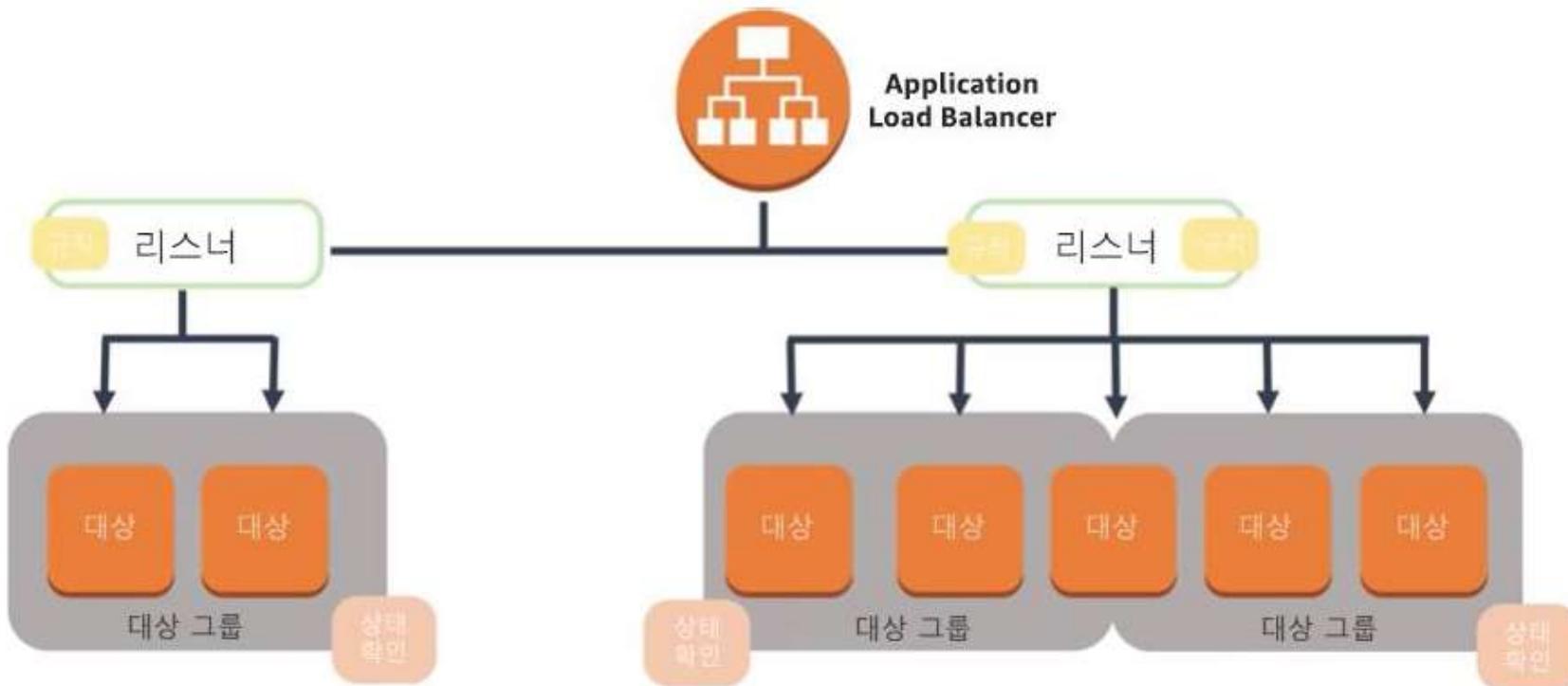
# AWS ELB

- **Application Load Balancer**
  - 로드 밸런서는 클라이언트에 대한 단일 접점 역할을 수행
  - 로드 밸런서는 여러 가용 영역에서 EC2 인스턴스 같은 여러 대상에 수신 애플리케이션 트래픽을 분산하여 가용성 향상
- **Network Load Balancer**
  - 급격하고 변동이 심한 트래픽 패턴
  - 가용 영역당 하나의 고정 IP 주소
  - 매우 뛰어난 성능이 필요한 애플리케이션에 적합
- **Classic Load Balancer**
  - 단일 지점을 통해 서버에 액세스
  - 애플리케이션 환경을 분리
  - 고가용성과 내결함성을 제공
  - 탄력성과 확장성 향상

# AWS ELB



# AWS ELB

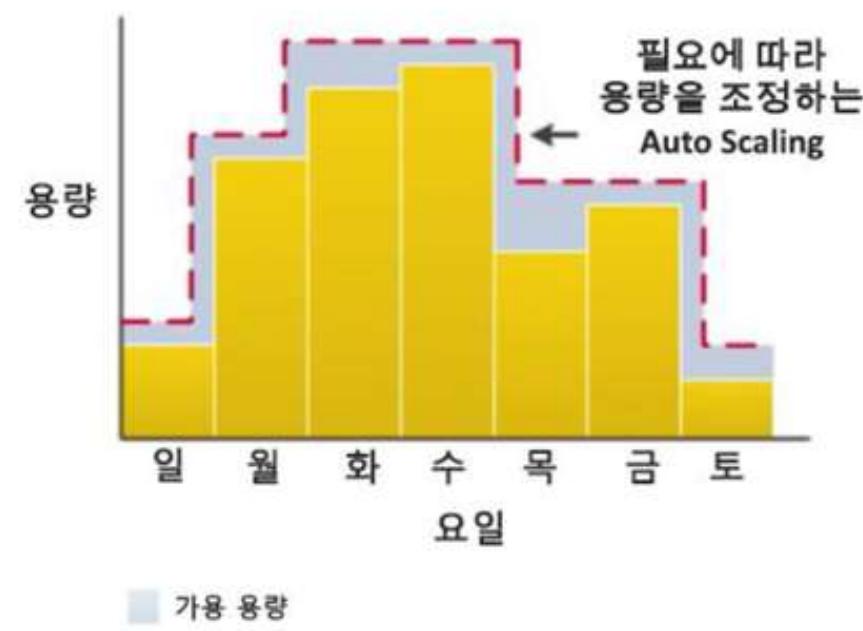
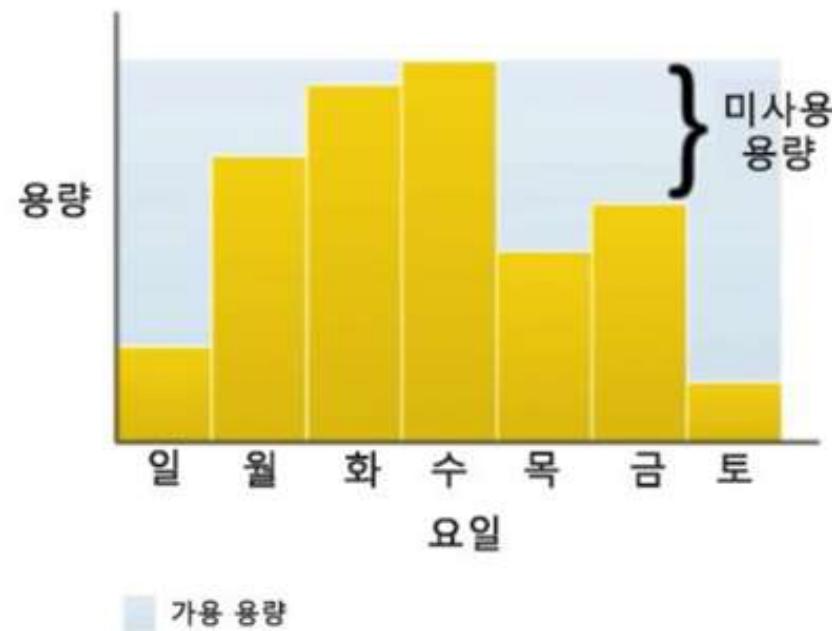


# Autoscaling

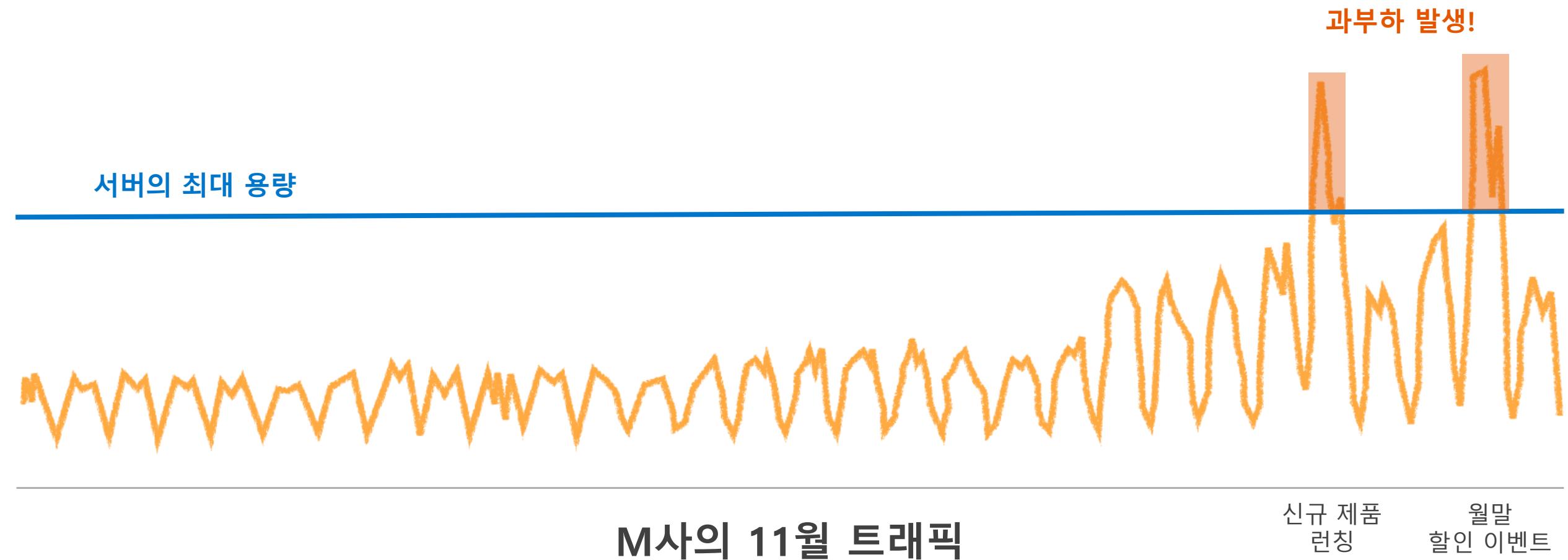
# Autoscaling

## Autoscaling 이란 ?

- Application 의 Load를 처리하는데 적절한 수의 EC2 Instance를 유지하도록 도움을 주는 기능



# Autoscaling



# Autoscaling

서버의 최대 용량

76% 의 낭비 자원 발생

M사의 11월 트래픽

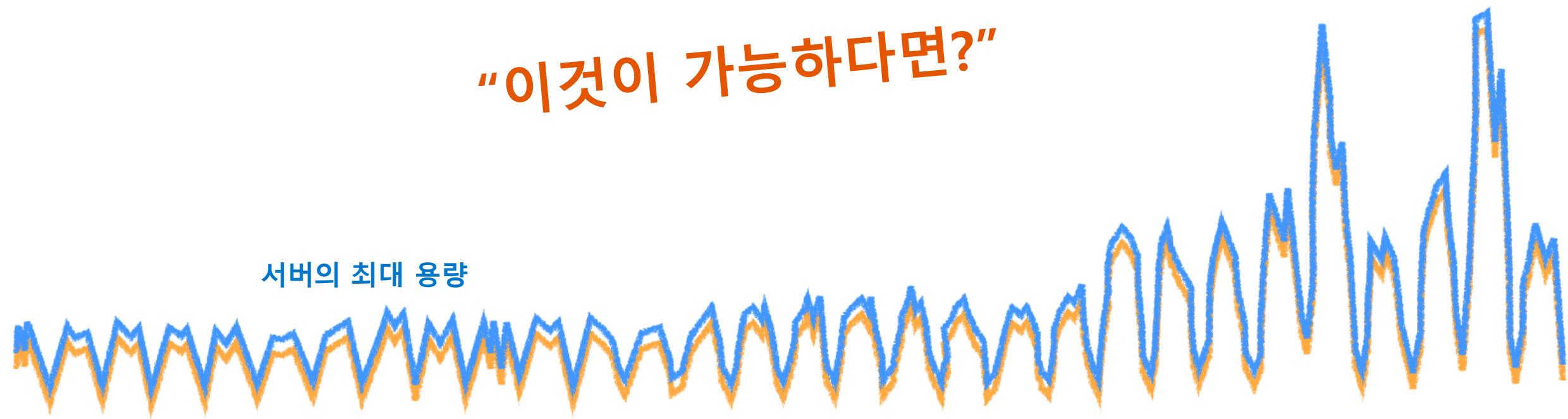
신규  
제품  
런칭

월말  
할인 이벤트

# Autoscaling

“이것이 가능하다면?”

서버의 최대 용량



M사의 11월 트래픽

신규  
제품  
런칭

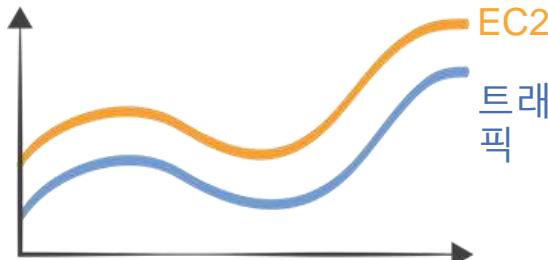
월말  
할인 이벤트

# Autoscaling

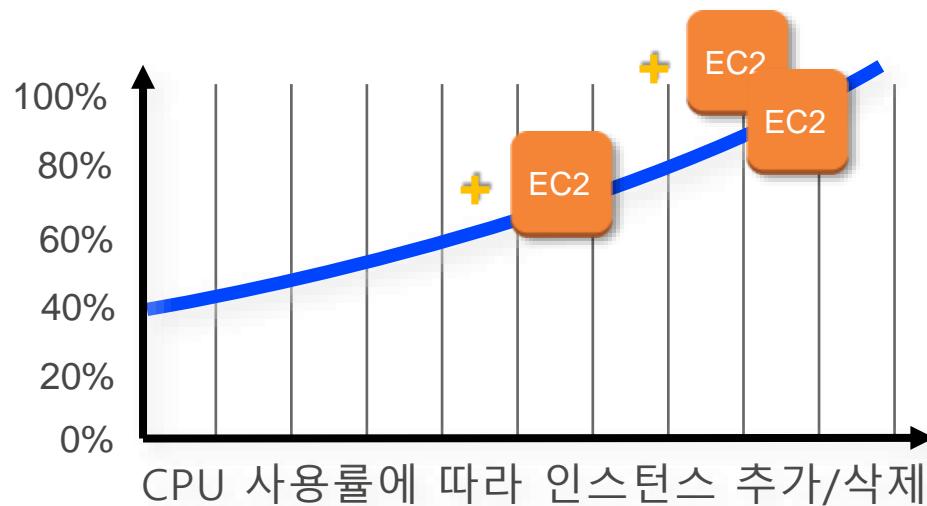


## Auto Scaling

“서버 자동 확장/축소”



- ✓ 평균 CPU 사용률이 **80~100%** 일때 인스턴스 **2개 추가**
- ✓ 평균 CPU 사용률이 **60~80%** 일때 인스턴스 **1개 추가**
- ✓ 평균 CPU 사용률이 **20~40%** 일때 인스턴스 **1개 제거**
- ✓ 평균 CPU 사용률이 **0~20%** 일때 인스턴스 **2개 제거**
- ❖ 최소 : 인스턴스 5개 (예시)
- ❖ 최대 : 인스턴스 20개 (예시)



# Autoscaling

## 리소스 성능 모니터링

- **Amazon CloudWatch**에서는 성능을 모니터링
- **Auto Scaling**에서는 EC2 인스턴스를 추가하거나 제거
  - **EC2 Auto Scaling**

## 중요한 질문

- 어떻게 내 워크로드가 변동하는 성능 요구 사항을 충족하기에 충분한 **EC2** 리소스를 확보할 수 있는가?

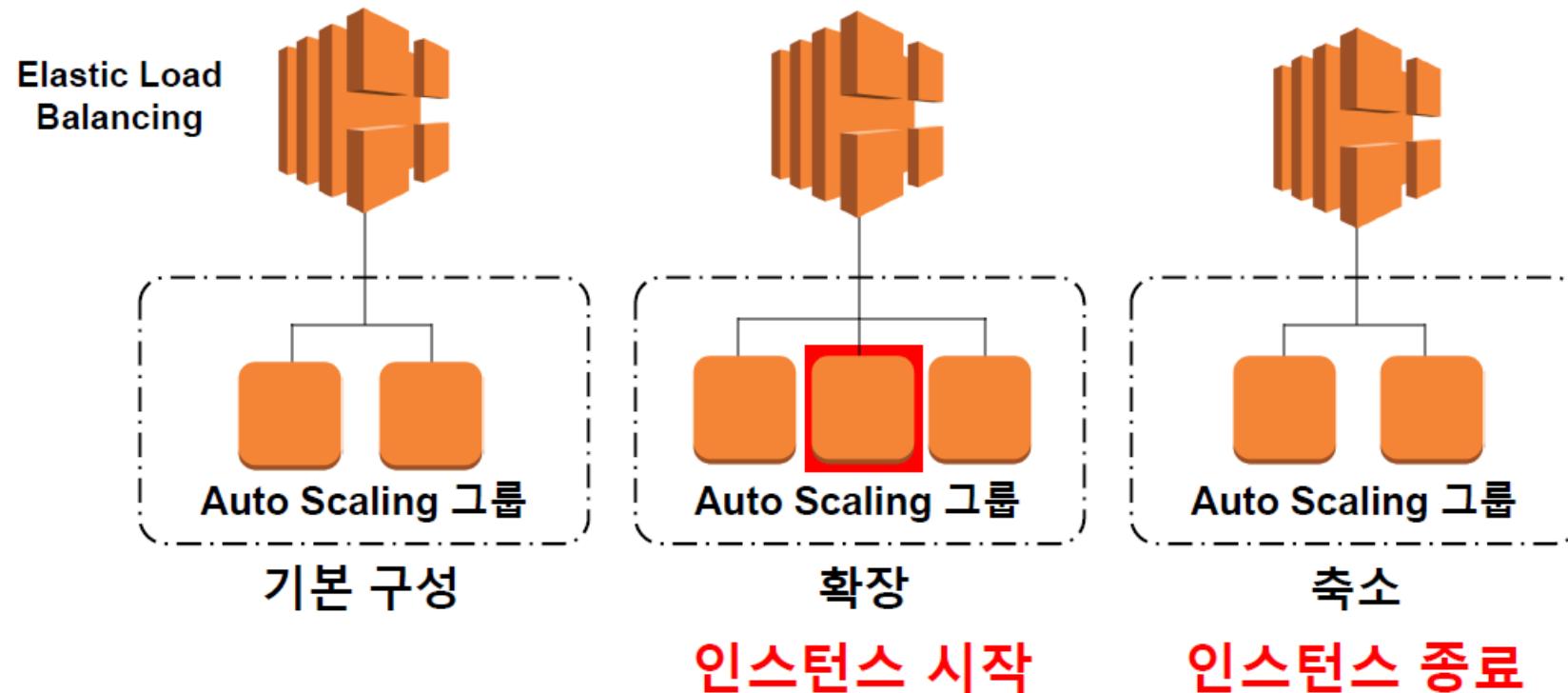
확장성

- 어떻게 **EC2** 리소스 프로비저닝이 필요에 따라 이루어지도록 자동화할 수 있는가?

자동화

# Autoscaling

## 확장 및 축소



# Autoscaling

## Auto Scaling 구성 요소

- 시작 구성
- Auto Scaling 그룹
- Auto Scaling 정책

### 시작 구성: 무엇이 조정될까요?

- 시작 설정
  - AMI
  - 인스턴스 유형
  - 보안 그룹
  - 역할

# Autoscaling

## Auto Scaling 구성 요소

### Auto Scaling 그룹: 언제 조정이 이루어질까요?

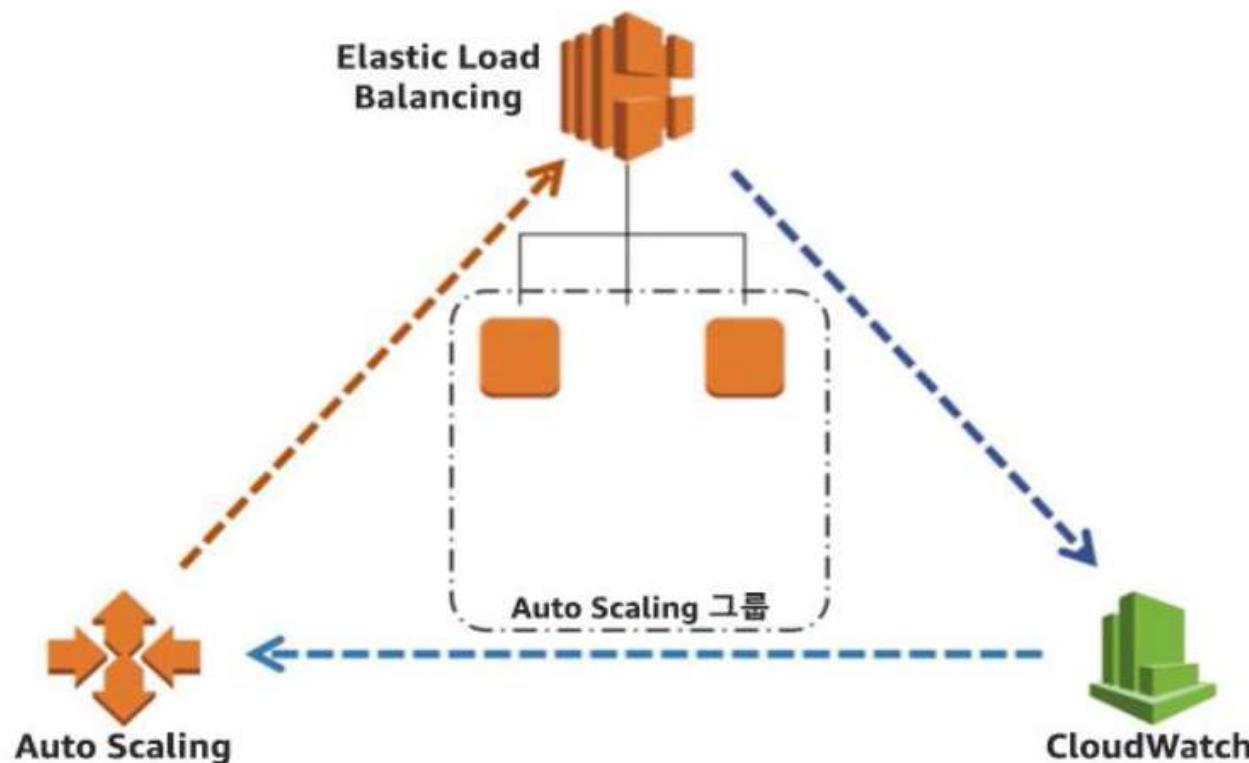
- 정책 설정
  - 일정 예약
  - 온디맨드
  - 확장 정책
  - 축소 정책

### Auto Scaling 그룹: 어디에서 조정이 이루어질까요?

- 배포 설정
  - VPC 및 서브넷
  - 로드 밸런서
  - 최소 인스턴스
  - 최대 인스턴스
  - 원하는 용량

# Autoscaling

## 동적 Auto Scaling



# Autoscaling

## Auto Scaling용 CloudWatch 경보



Whenever: CPUUtilization  
is:  $\geq$  80  
for: 1 consecutive period(s)

AutoScaling Action

Whenever this alarm: State is ALARM

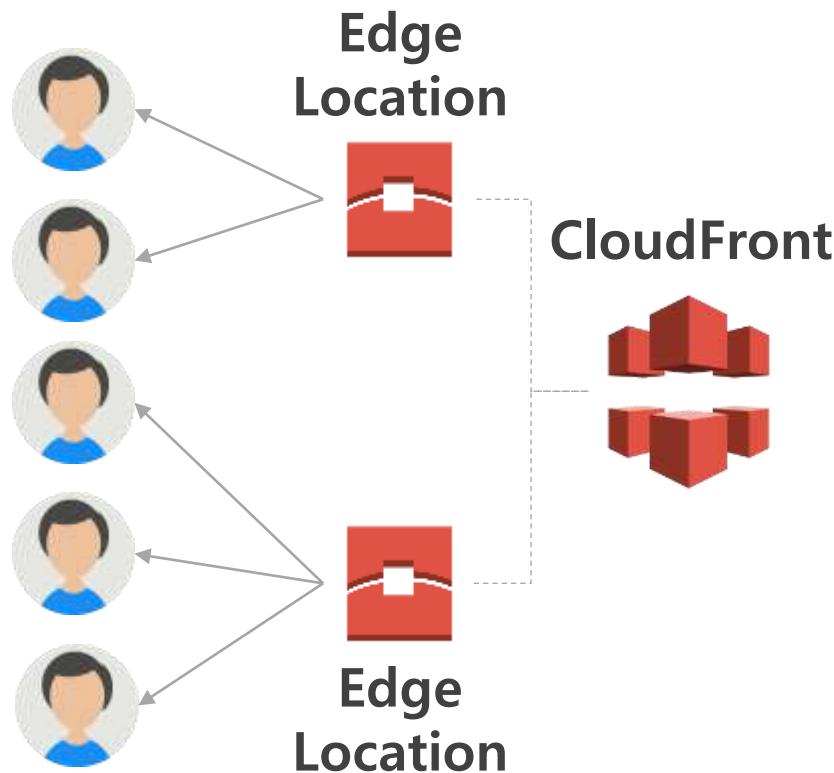
From resource type: AutoScaling

From the: IREASG

Take this action: Increase Group Size - Add 2 instances

A screenshot of the AWS CloudWatch Metrics Insights interface. It shows a configuration for an Auto Scaling action triggered by a CloudWatch Metrics alarm. The alarm triggers whenever the CPUUtilization metric is greater than or equal to 80 for one consecutive period. The action is to increase the group size by adding two instances for any Auto Scaling resource named IREASG whenever the alarm state is ALARM.

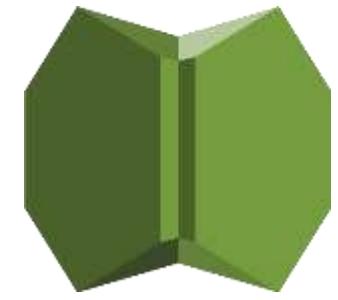
# 참고 : CloudFront



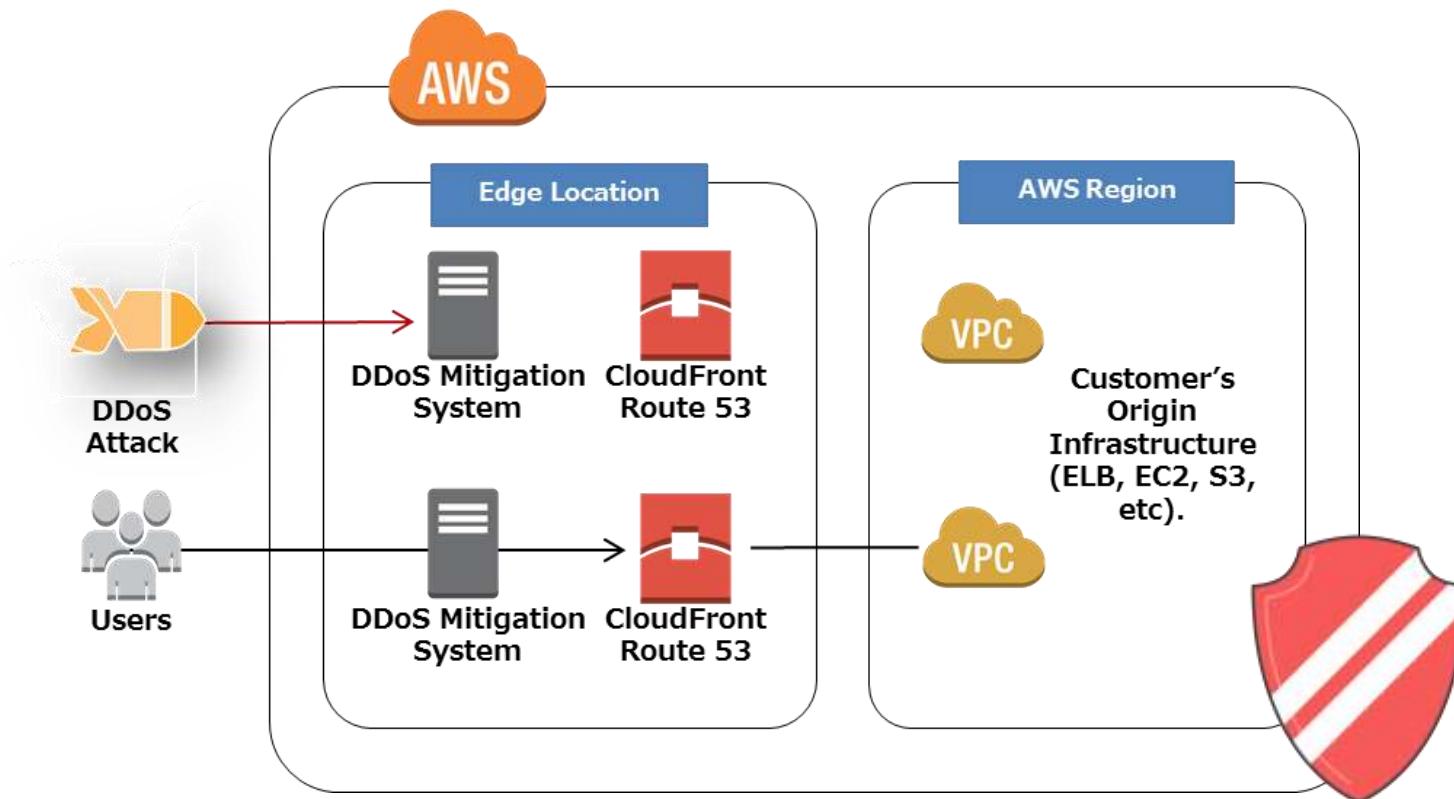
- 컨텐츠 (이미지, HTML 등)를 캐싱하여 성능 가속
- 전 세계 113개 엣지 로케이션: 글로벌 서비스
- AWS 서비스 → CloudFront 간 데이터 전송 무료
- 글로벌 고속 백본 네트워크 확보
- DDoS 방어 무료 제공 (AWS Shield Standard)

# 참고 : Amazon CloudFront: Layer 3/4 DDoS 방어 제공

“AWS Shield Standard에 의한 L3 / L4 DDoS 보호는 추가 비용 없이 포함”



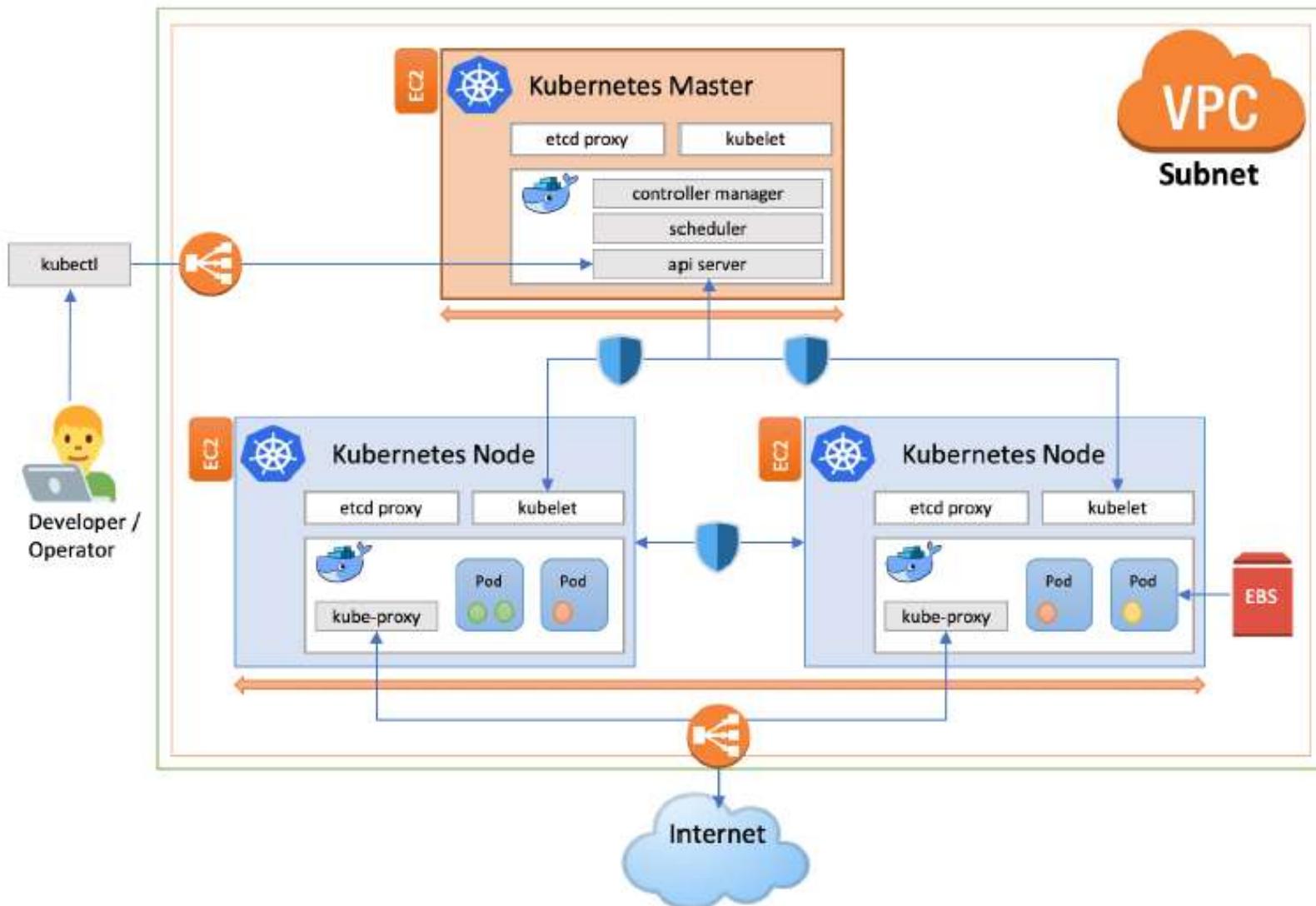
AWS  
Shield  
Standard



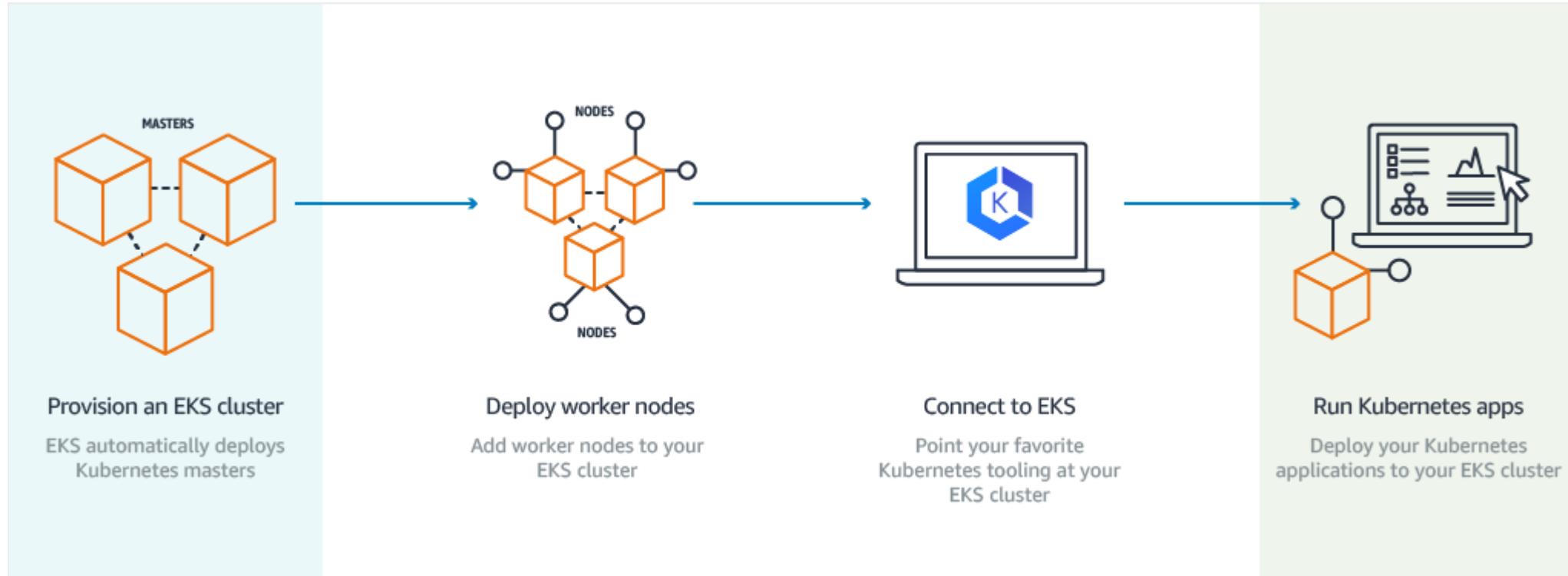
# **Part III**

# **AWS EKS**

# AWS에서의 쿠버네티스 구성 예

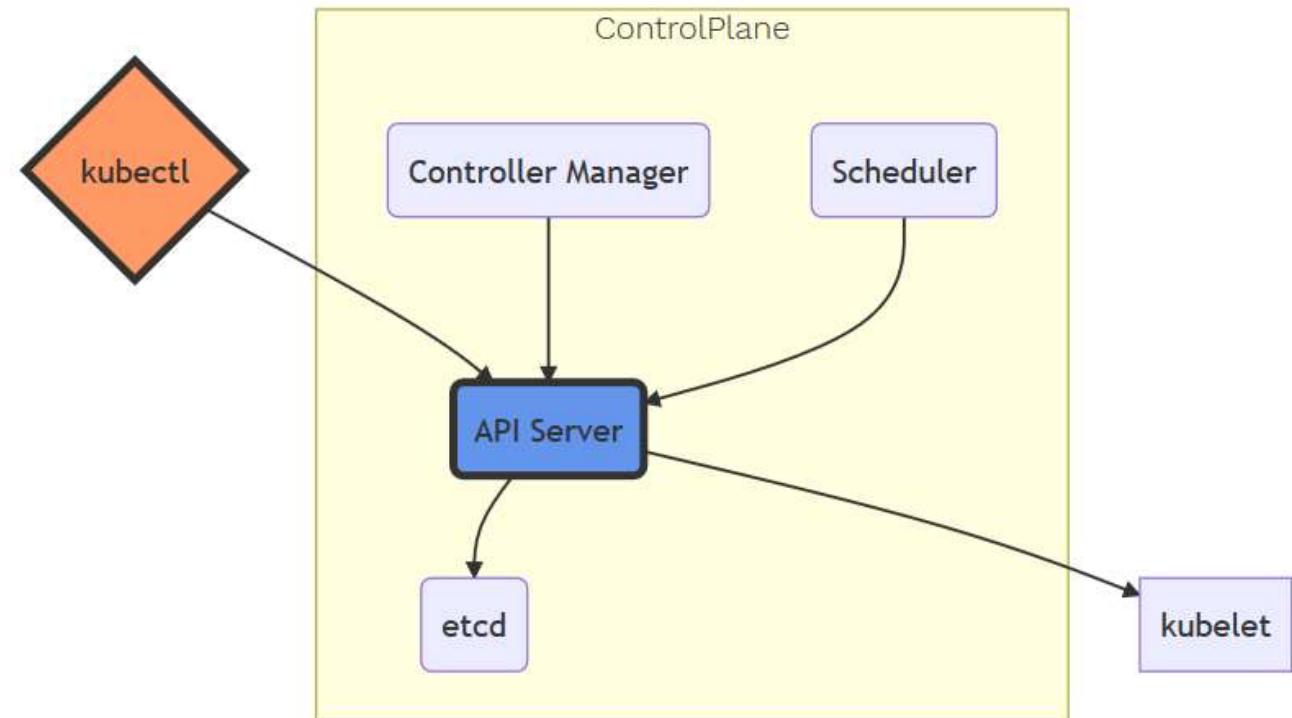


# AWS EKS 작동 방식



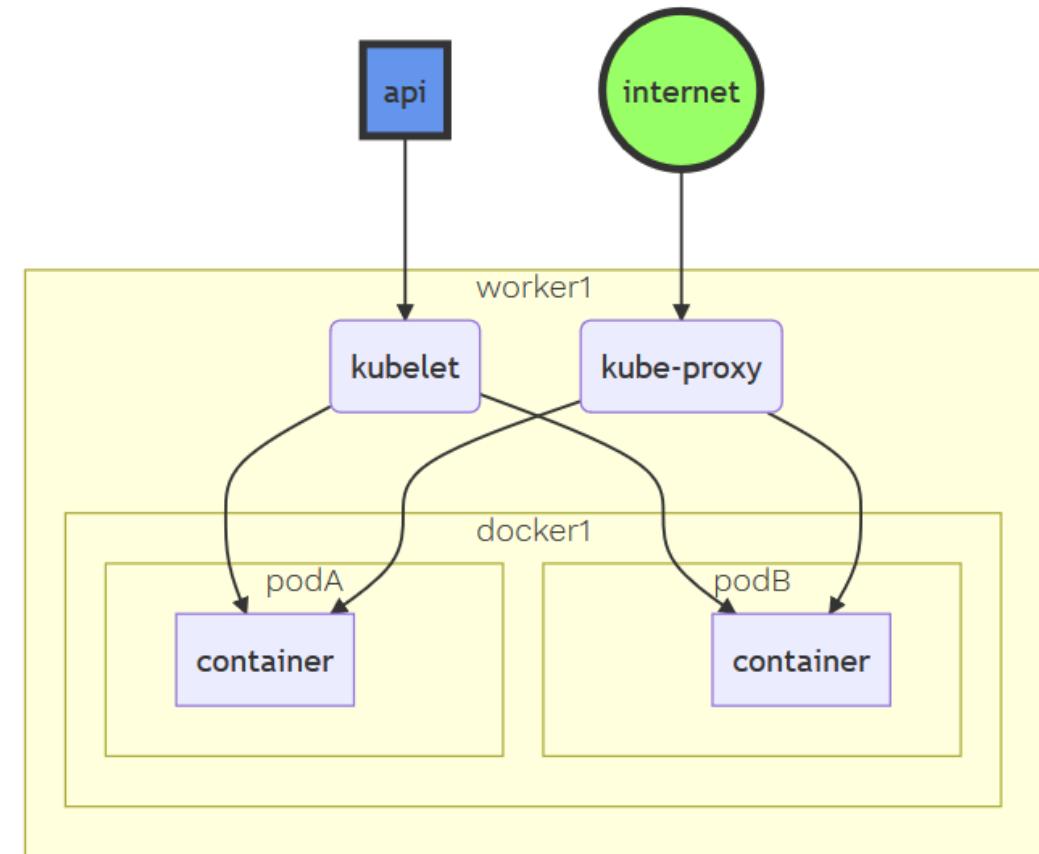
# EKS 제어 플레인

- 고 가용성을 내재한 단일 테넌트 인프라스트럭처
- NLB를 활용한 부하분산
- 컨트롤 플레인 로그
- API 서버, Audit, Controller Manager, Authenticator, Scheduler



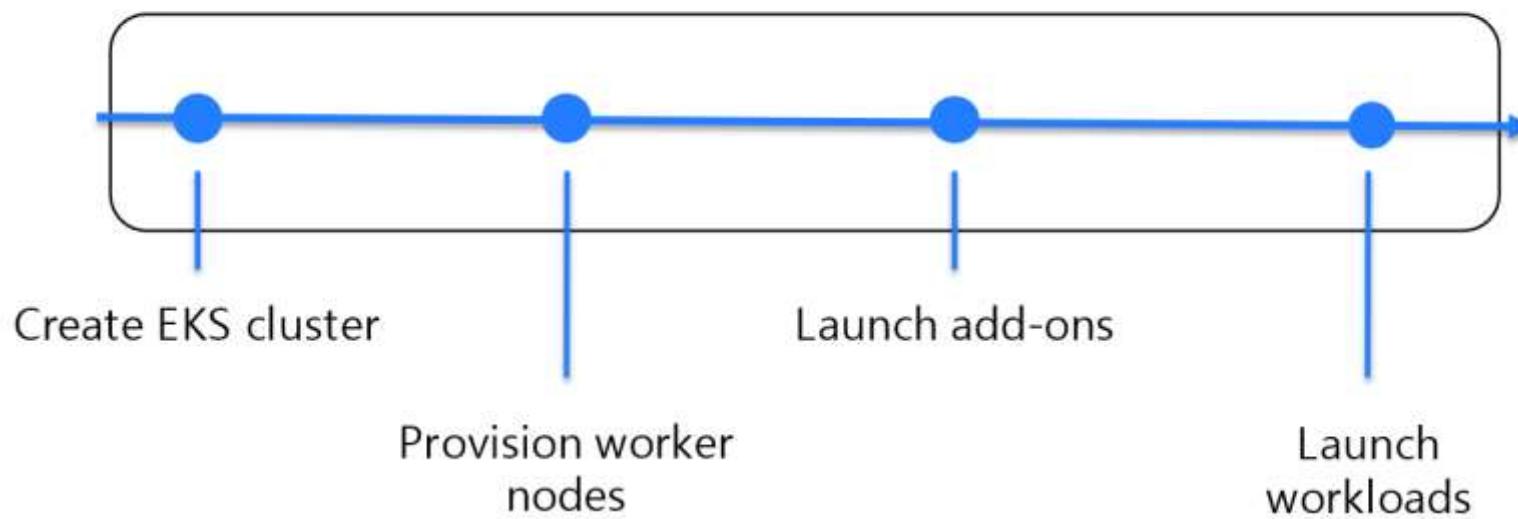
# AWS EKS 작동 방식

- 다양한 인스턴스 유형 및 가격정책
  - M5, C5, P2, P3 GPU, Spot or Mixed
- AI/ML용 Amazon EKS
- EKS에 최적화된 GPU지원 AMI
- EKS AMI 빌드 스크립트
- 워크로드 특성에 적합한 인스턴스 적용

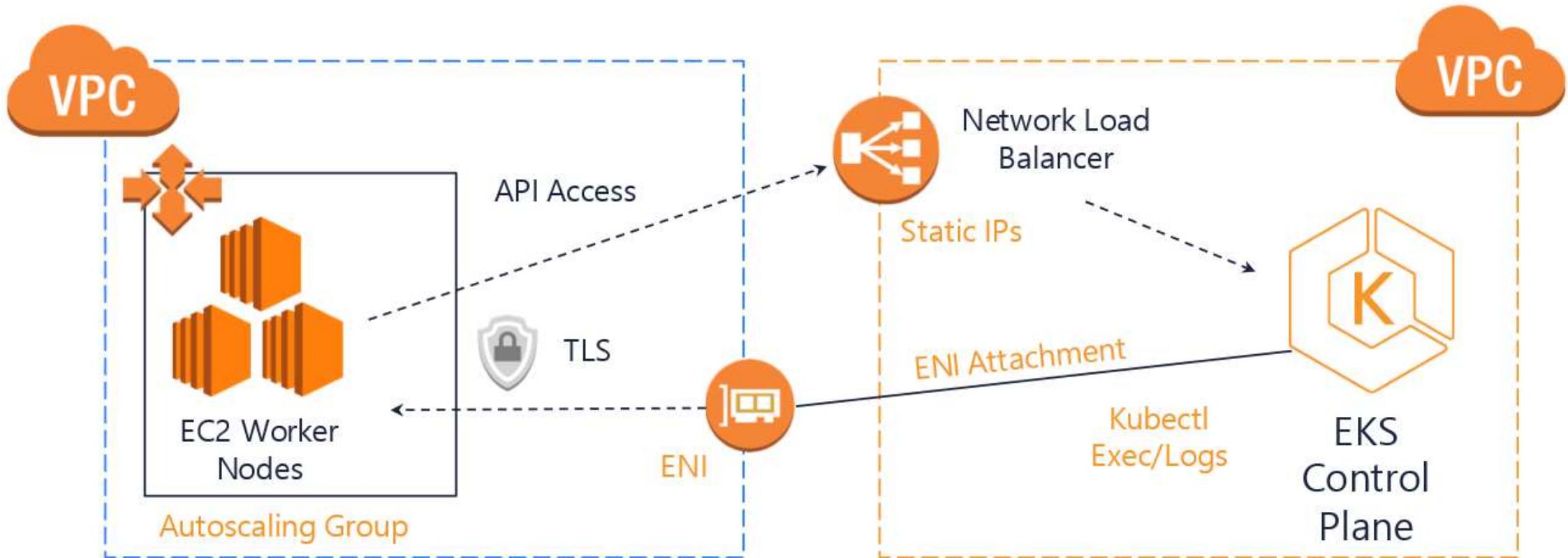


# EKS 클러스터 생성 워크플로우

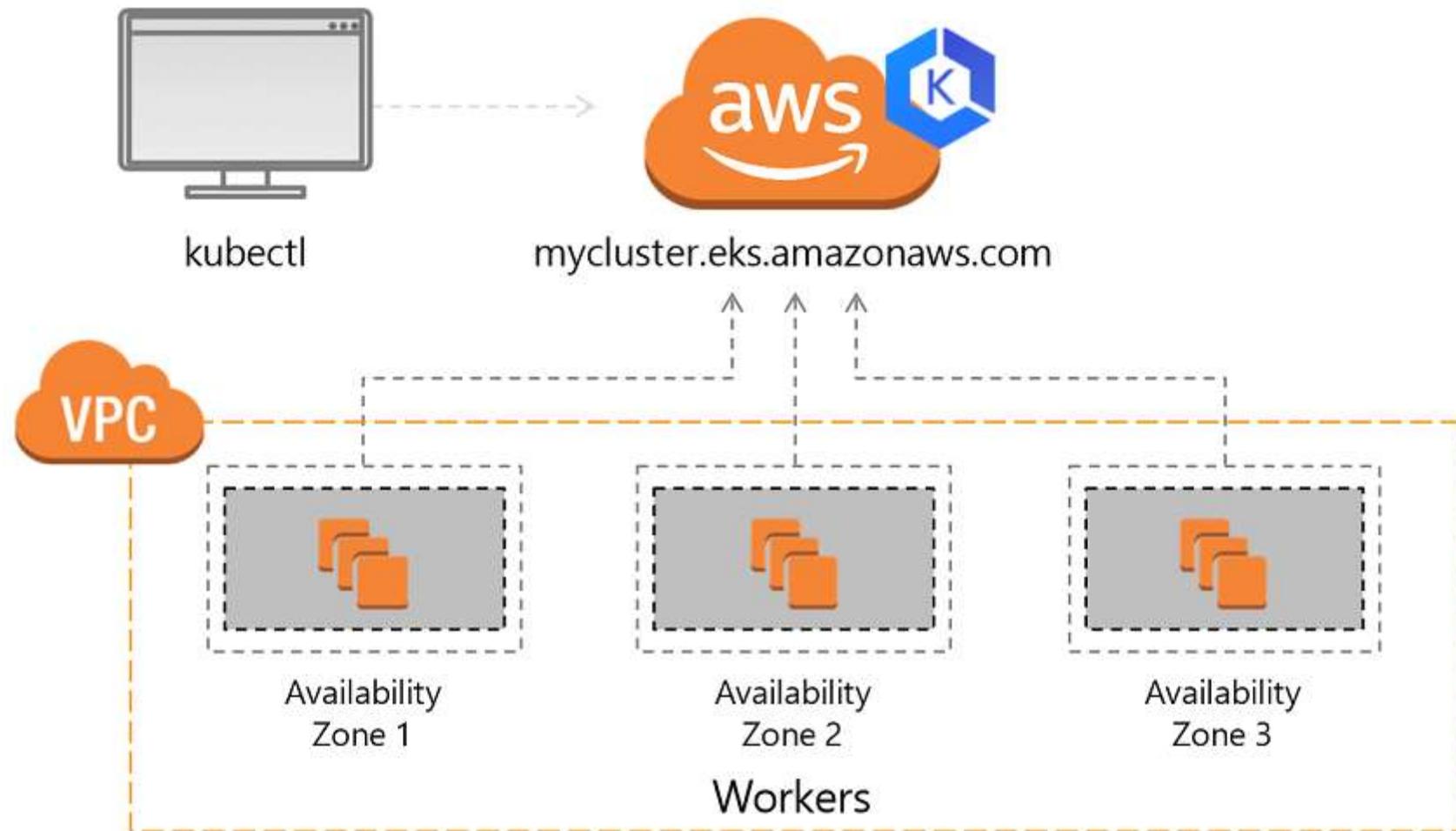
## EKS CLUSTER CREATION WORKFLOW



# EKS 제어 플레인과 Worker Node간의 통신



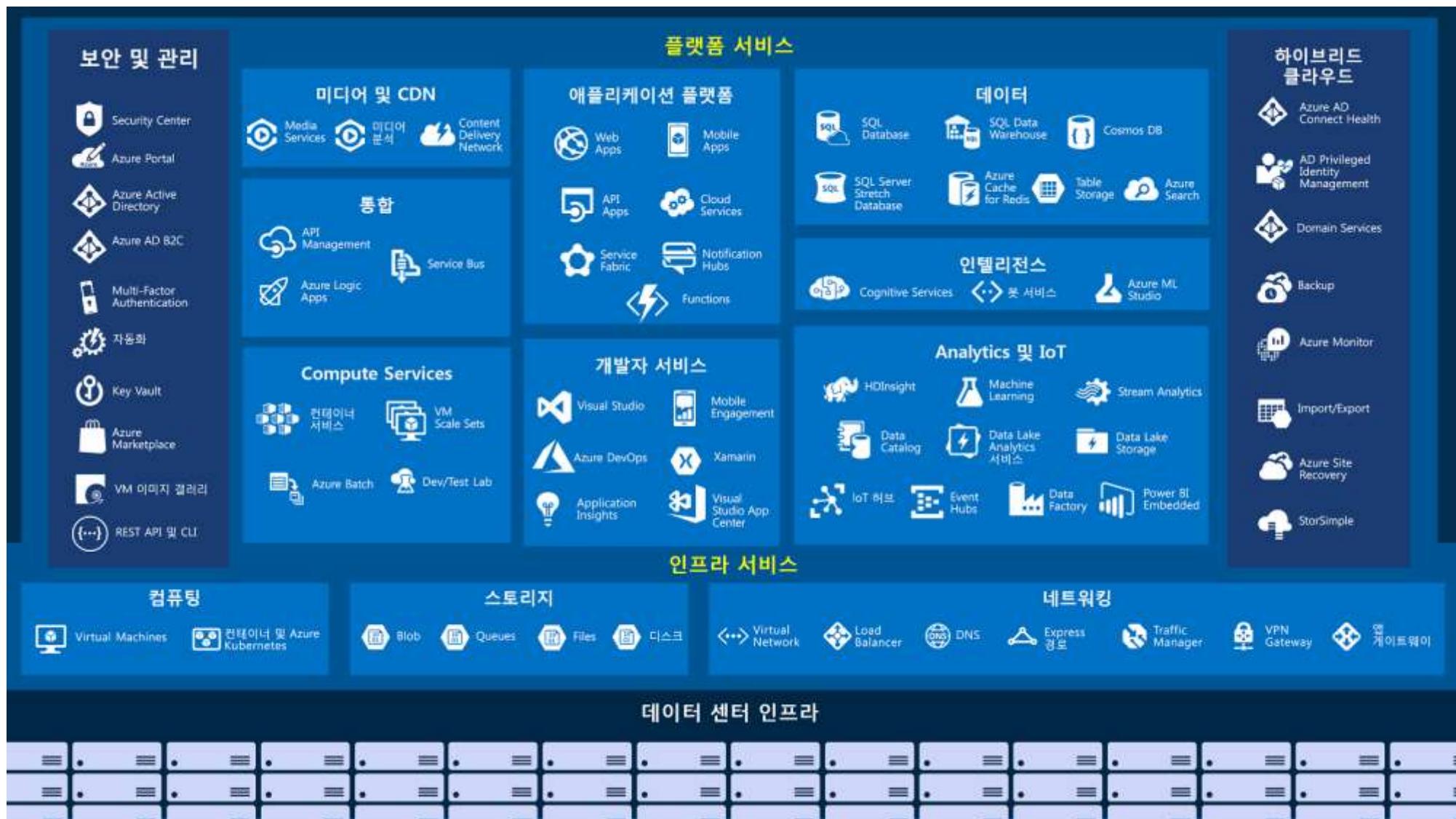
# EKS와 Kubectl



## Module 5

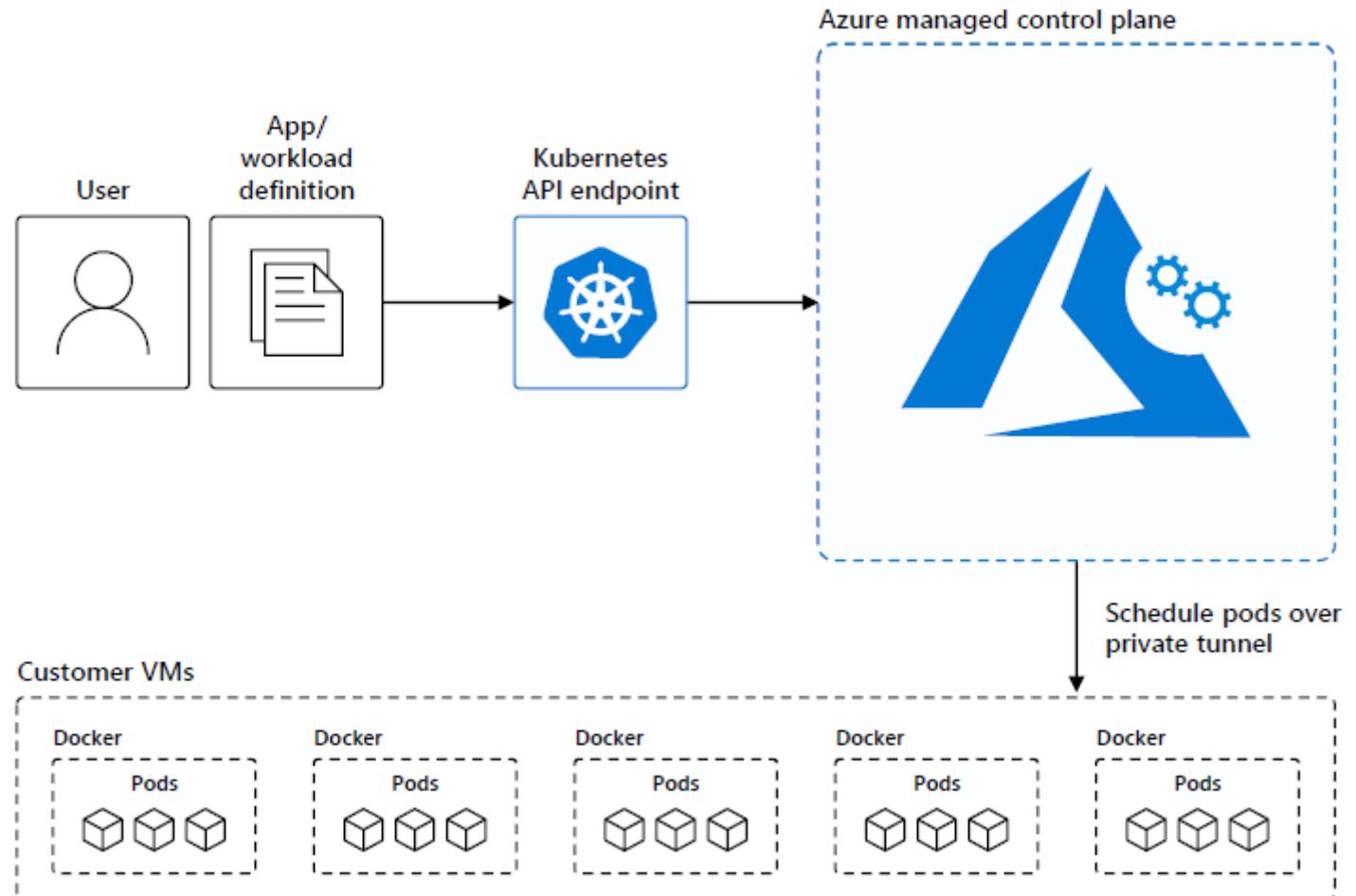
# MS Azure AKS를 이용한 Kubernetes 배포운영 이해와 실습

# Azure 서비스 포트폴리오



# How managed Kubernetes on Azure works

- Automated upgrades, patches
- High reliability, availability
- Easy, secure cluster scaling
- Self healing
- API server monitoring
- At no charge



# From infrastructure to innovation

Managed Kubernetes  
empowers you to do more

Focus on your containers  
and code, not the plumbing  
of them

Responsibilities	DIY with Kubernetes	Managed Kubernetes on Azure
Containerization		
Application iteration, debugging		
CI/CD		
Cluster hosting		
Cluster upgrade		
Patching		
Scaling		
Monitoring and logging		

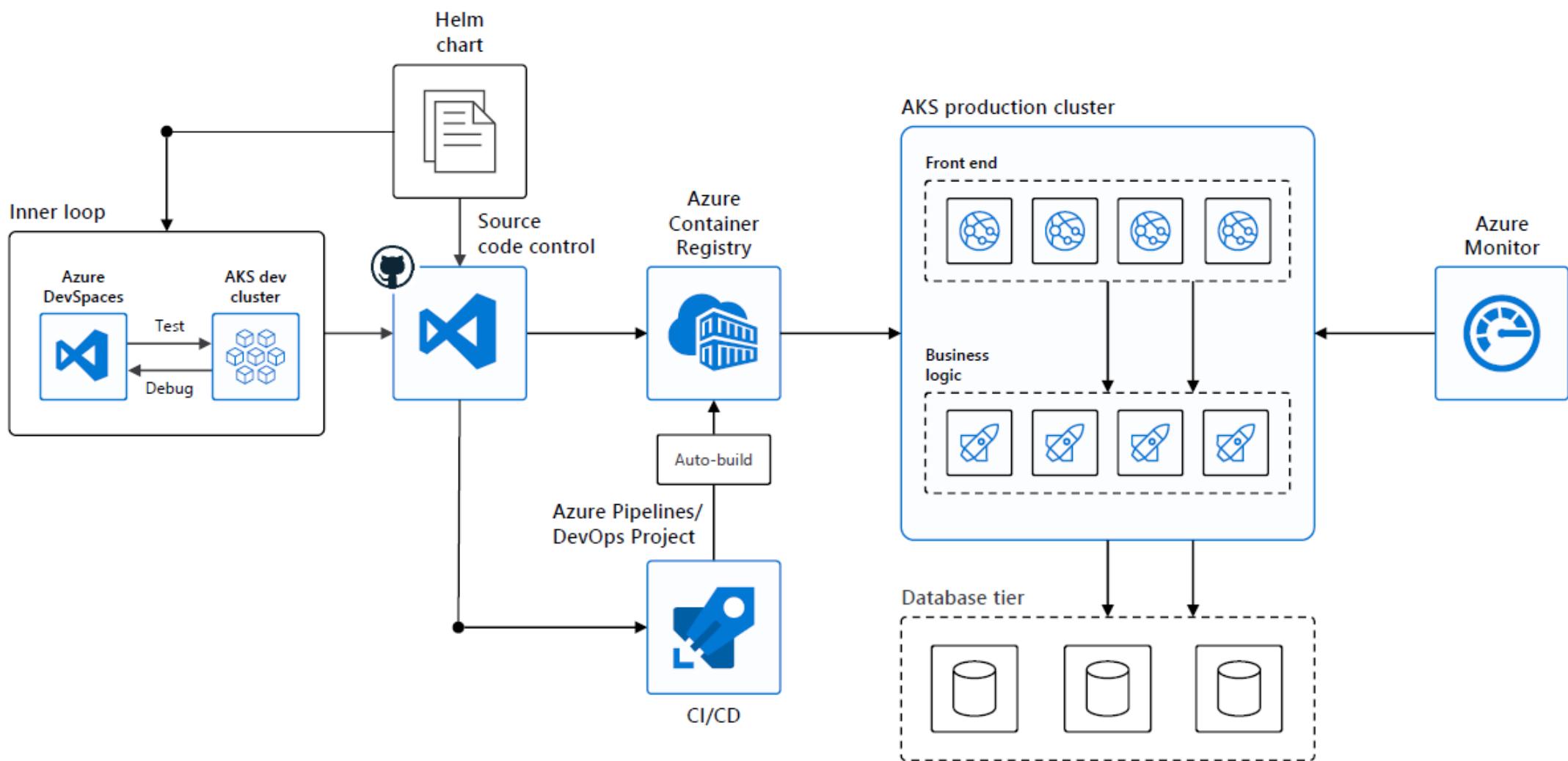
Customer      Microsoft

# Azure makes Kubernetes easy

Deploy and manage Kubernetes with ease

Task	The old way	With Azure
Create a cluster	Provision network and VMs Install dozens of system components including etcd Create and install certificates Register agent nodes with control plane	<code>az aks create</code>
Upgrade a cluster	Upgrade your master nodes Cordon/drain and upgrade worker nodes individually	<code>az aks upgrade</code>
Scale a cluster	Provision new VMs Install system components Register nodes with API server	<code>az aks scale</code>

# End to End Experience



# Secure your Kubernetes environment

## Compliance



Control access through  
AAD and RBAC



Safeguard keys and  
secrets with Key Vault



Secure network  
communications with  
VNET and CNI



Compliant Kubernetes  
service with  
certifications covering  
SOC, HIPAA, and PCI



# Scale and run with confidence



Built-in  
auto scaling



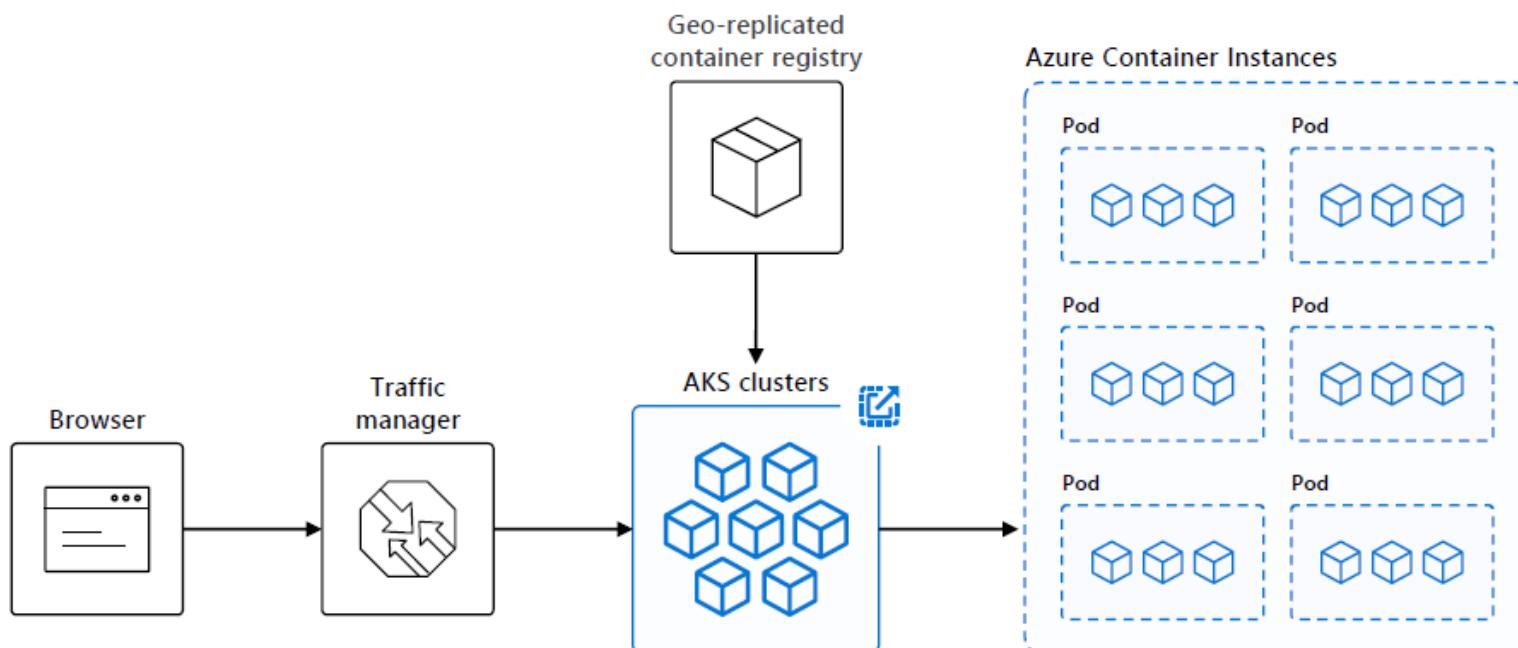
Global  
data center



Elastically burst  
using ACI

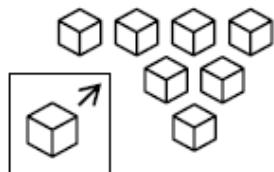


Geo-replicated  
container registry



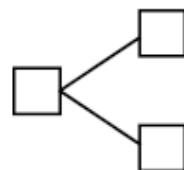
# Top scenarios for Kubernetes on Azure

## Lift and shift to containers



Cost saving  
without refactoring  
your app

## Microservices



Agility  
Faster application  
development

## Machine learning



Performance  
Low latency  
processing

## IoT

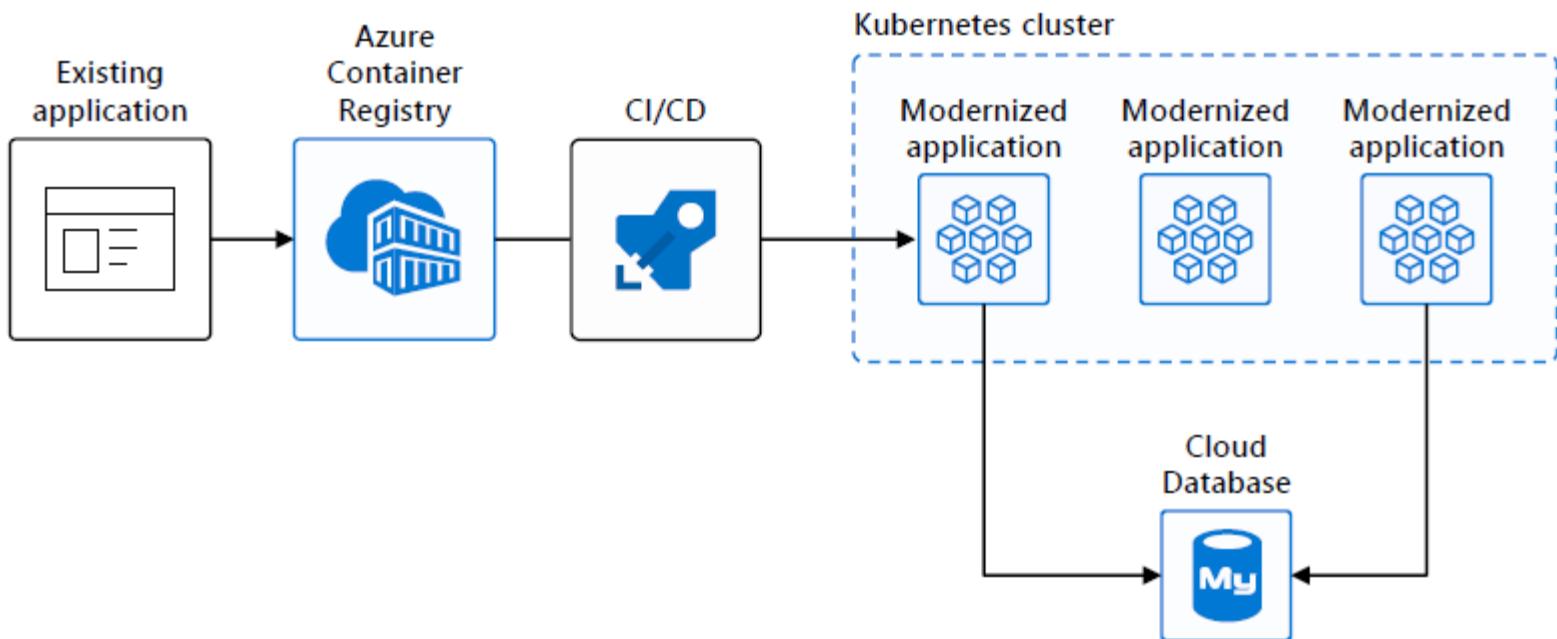


Portability  
Build once, run  
anywhere



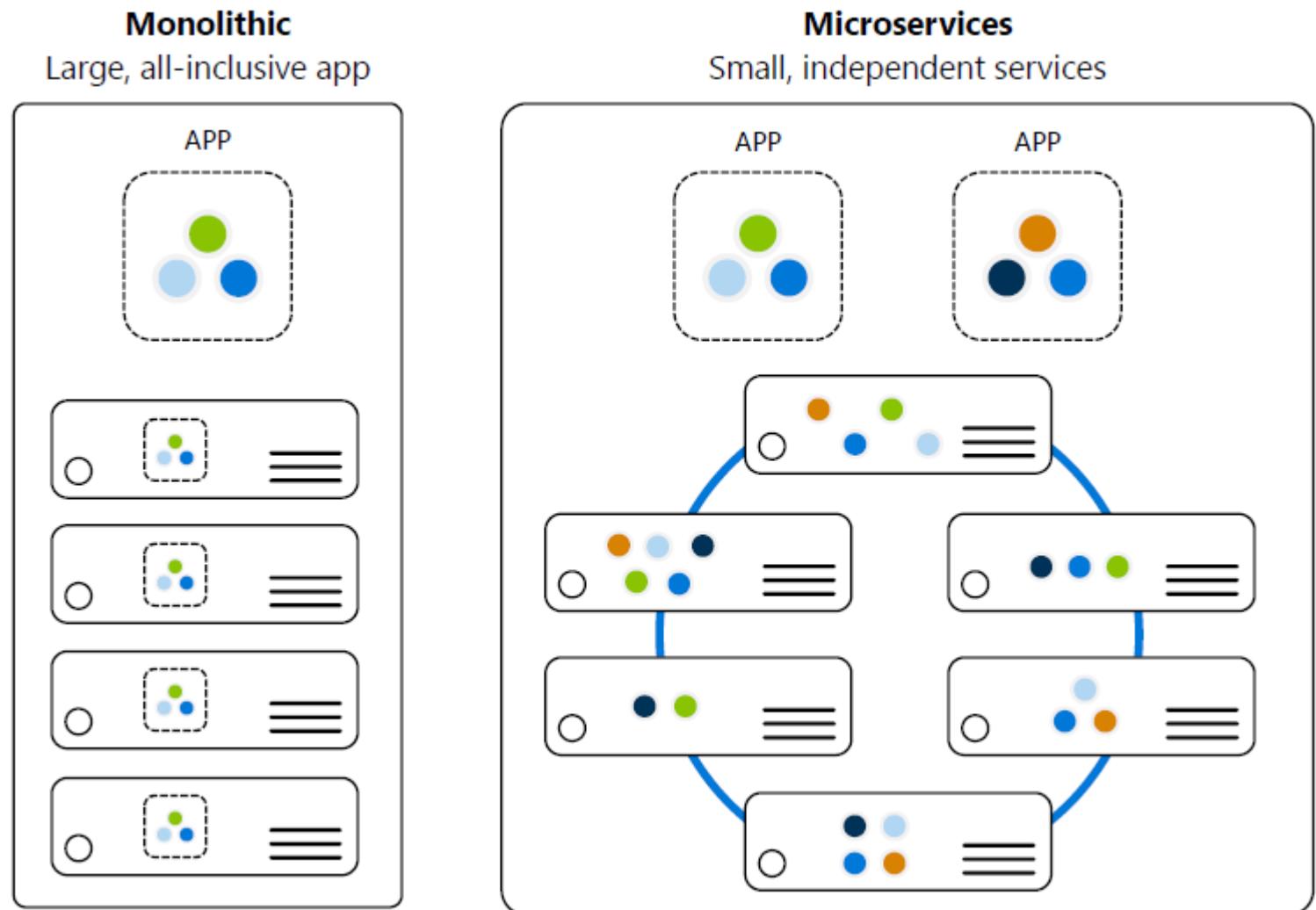
# App modernization without code changes

- Speed application deployments by using container technology
- Defend against infrastructure failures with container orchestration
- Increase agility with continuous integration and continuous delivery



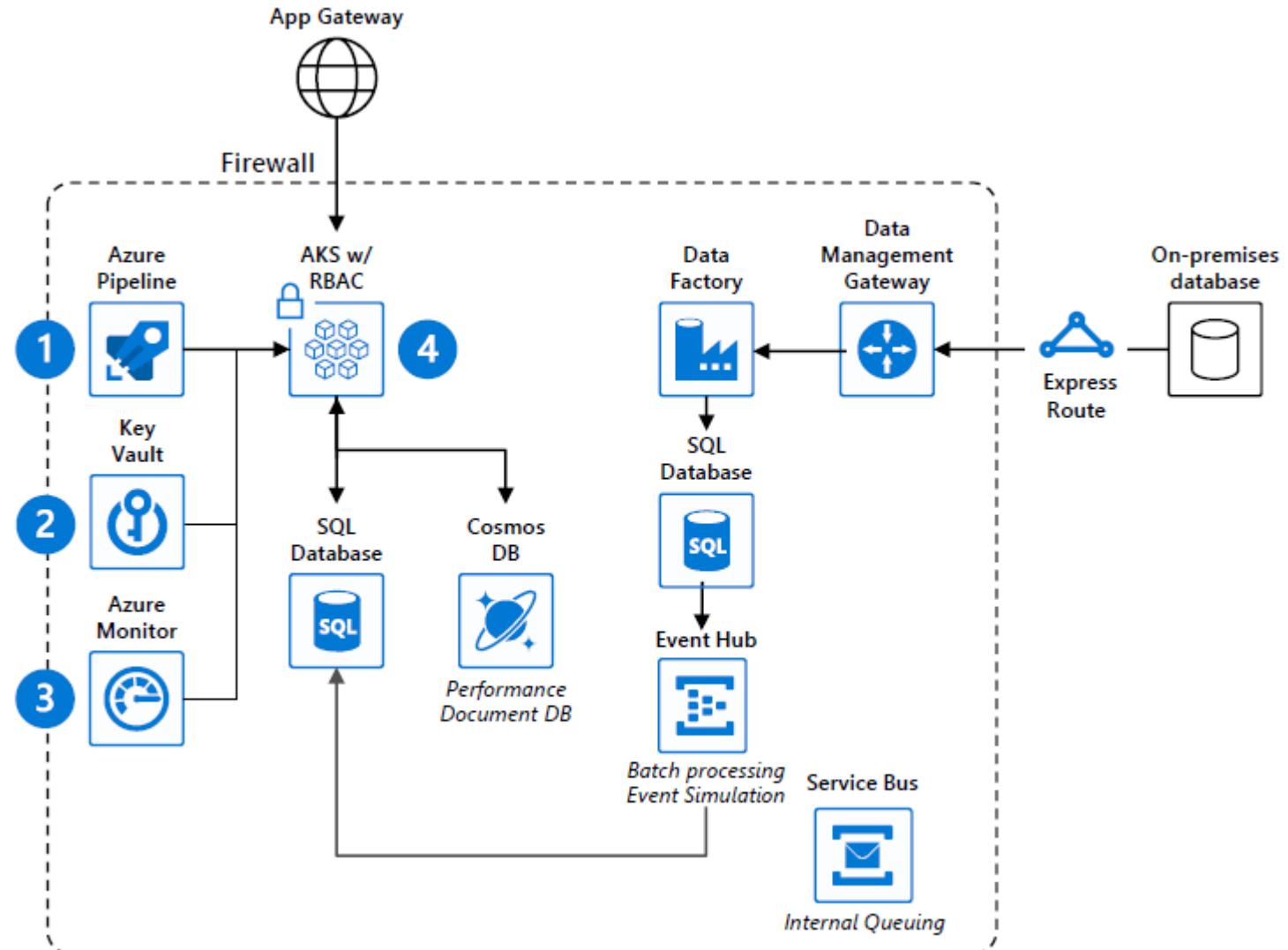
# Microservices: for faster app development

- Independent deployments
- Improved scale and resource utilization per service
- Smaller, focused teams



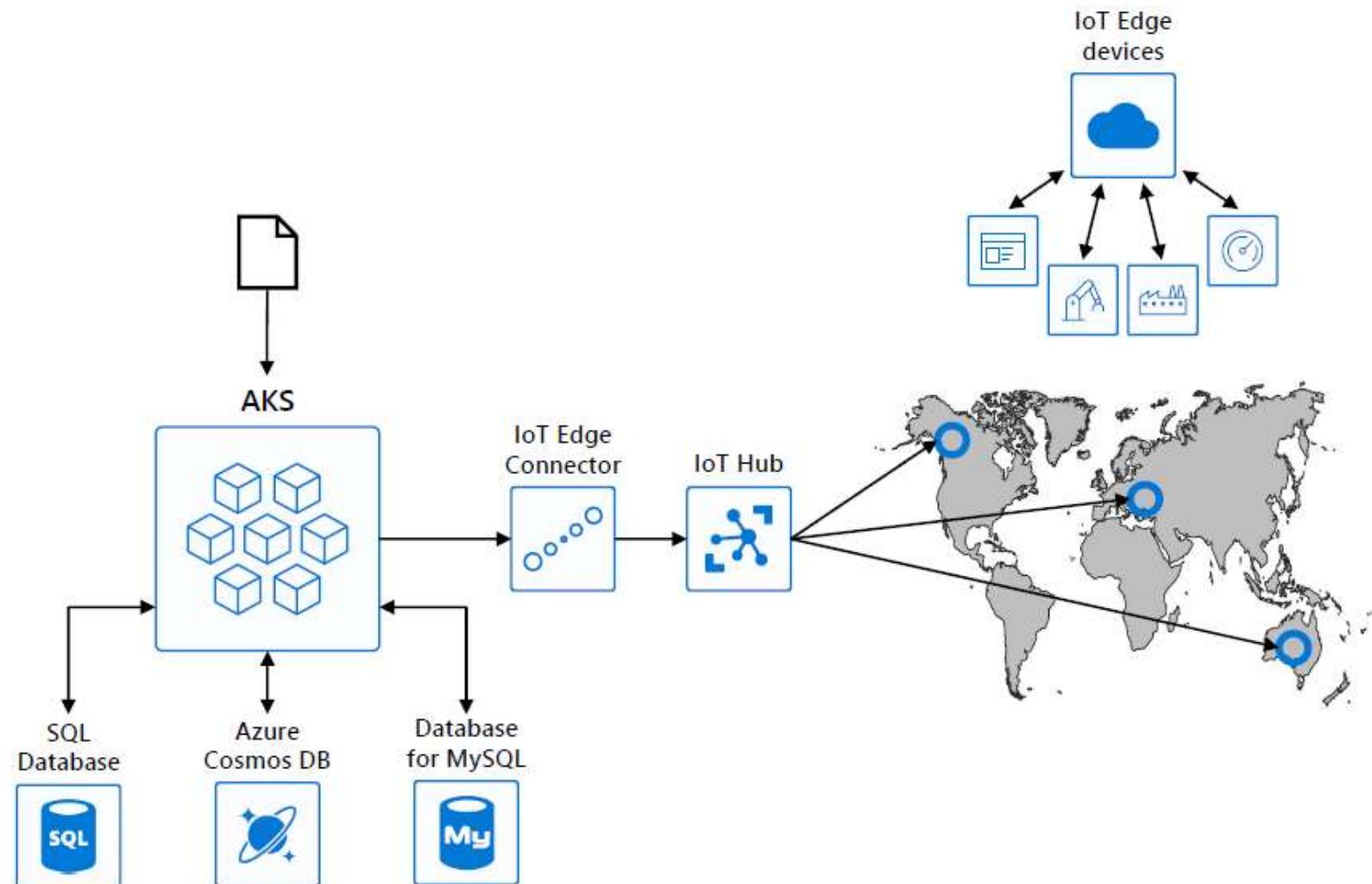
# Architectural approach

1. Azure Pipelines for automation and CI/CD pipelines; adding Terraform for further automation
2. Key Vault to secure secrets and for persistent configuration store
3. Azure Monitor for containers provides better logging, troubleshooting, with no direct container access
4. RBAC control for fine grained Kubernetes resources access control



# Example : Scalable Internet of Things solutions

- Portable code, runs anywhere
- Elastic scalability and manageability
- Quick deployment and high availability



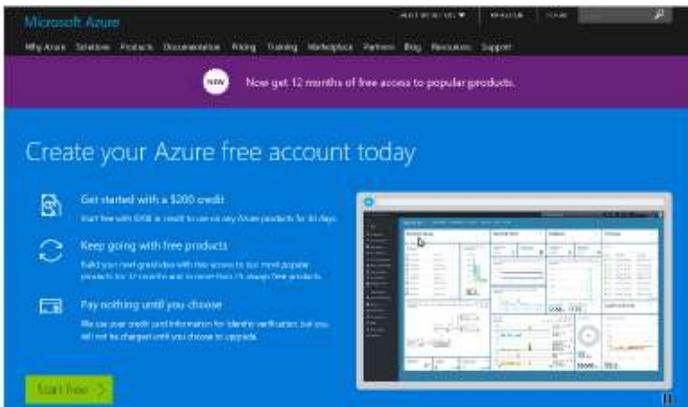
# Work how you want with opensource tools and APIs

	Development	DevOps	Monitoring	Networking	Storage	Security
Take advantage of services and tools in the Kubernetes ecosystem	 	     	     	 	 	   RBAC
...or... Leverage growing Azure support		 Azure DevOps 				 Azure Container Registry  AAD  Key Vault

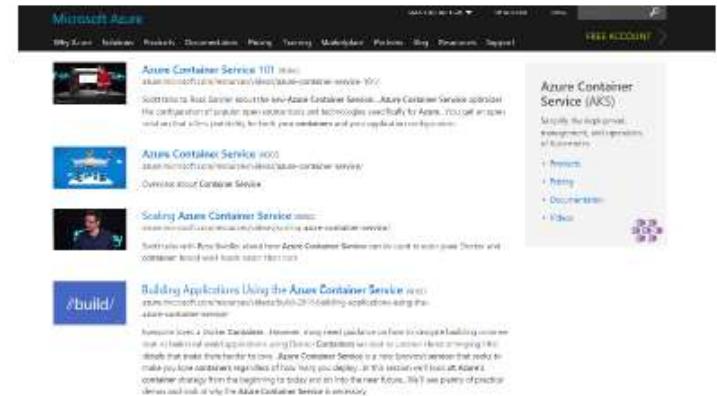
# AKS resources

- Azure Kubernetes Service (AKS)
- Containers on Azure pitch deck
- Smart Hotel 360 Demo
- Documentation resources
- Ebook for distributed systems
- Distributed system HoL
- AKS HoL

Sign up for a free Azure account



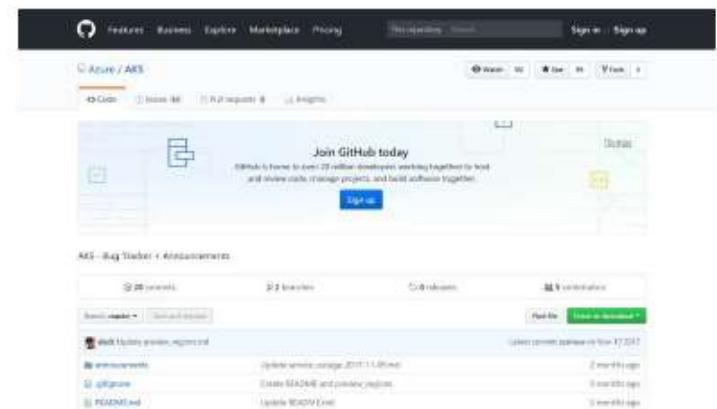
Check out the Azure container videos page



Hone your skills with Azure training



Get the code from GitHub



## Module 6

# Agile DevOps와 MSA 이해와 환경 구성

# **Part I**

# **Agile Culture**

# 애자일(Agile)이란 ?

## 정의

- 목표 : 고객이 원하는 요구사항(customer needs)을 정확하게 파악하고, 짧은 주기, 반복적으로(regularly) 반영하여 퀄리티(quality)가 높은 애플리케이션을 배포하는 것
- 위 목표를 성취하기 위한 일하는 방식과 방법론을 애자일 이라고 부름

## 장점

- 1주일 단위로 고객의 피드백 수집과 우선순위화의 변경이 일어나기 때문에, 잘못된 방향으로 진행되는 리스크를 Zero로 만들 수 있음
- 즉각적인 반응을 볼 수 있기 때문에, 참여하는 사람들의 Ownership기반의 일하는 즐거움을 찾을 수 있는 것이 목표

## 유의 사항

- 페어(Pair) 환경, 커뮤니케이션 툴, 자동 배포, 개발 플랫폼 등의 디지털 환경에 대한 투자가 반드시 필요
- 한꺼번에 많은 팀을 시작하는 것은 불가능. 1-2개의 팀을 제대로 진행해 보는 것이 필요
- Top-Down의 강제성이 아닌 자원(Volunteer)하여 참여하고 싶도록 만드는 인센티브에 대한 고민
- 애자일 액티비티의 모니터링 대시보드 필요

# 데브옵스(DevOps)란 ?

## 정의

- 개발자에게는 **셀프서비스 툴의 권한**을 부여하여 Time-To-Market을 극대화하고 (보안의 희생 없이), 운영자는 인프라를 소프트웨어적인 관점에서 바라보아 **최대한의 자동화를 추구하는 것**
- CI/CD 파이프라인 구성, Infra as Code, IAM 등의 업무 환경 구성이 요구됨

## 장점

- 자동화를 통해 “요청”에서부터 “반영” 까지 걸리는 시간을 **최소화** (Value Delivery Time 최적화)
- 개발자/운영자 간의 분리된 목표로 움직이는 것이 아닌 **협업의 문화**

## 유의 사항

- 보안/규정의 희생 없이 최대한 자동화 하는 역량의 내재화 필요 (Speed와 Security 모두를 추구)
- 개발팀(스피드우선)과 운영팀(안정성우선)의 대립을 협업으로 바꾸기 위한 공통적 관심사/목표 도출이 필요
- Infra as Code, Immutable Deploy, API Based Operation 등 Skill Set의 내재화가 필요

# 애자일 도입 예시 (일주일 단위 반복)

애자일 단계

고객 요구사항 수집



- CS(고객서비스) 데이터 및 고객 피드백을 바탕으로 모든 요구사항을 수집

스토리  
분류



- 프로젝트의 크기를 가늠할 수 있는 첫 번째 단계

스토리  
점수 평가



- 작업의 난이도에 따라 포인트를 부여함
- Bug 및 Chore에 대해서는 포인트 부여 안함
- 스토리의 시급성에 따라 우선순위를 다같이 정함
- 우선순위를 정하면 한주에 처리 할 수 있는 “팀의 속도”에 따라 자동으로 종료 일정이 계산되게 됨

스토리  
우선순위화



디자인 & 개발



- 짹 활동 (개발자, PM, 디자이너 포함)을 통해 협업
- 테스트/QA를 개발과 병렬적으로 진행 해야함

수용 or 거절



- 개발/디자인이 완료된 결과물을 PM과 개발자가 동시에 Accept를 하면 완료
- 원하는 결과물이 아닌 경우 수정 요청 혹은 Reject

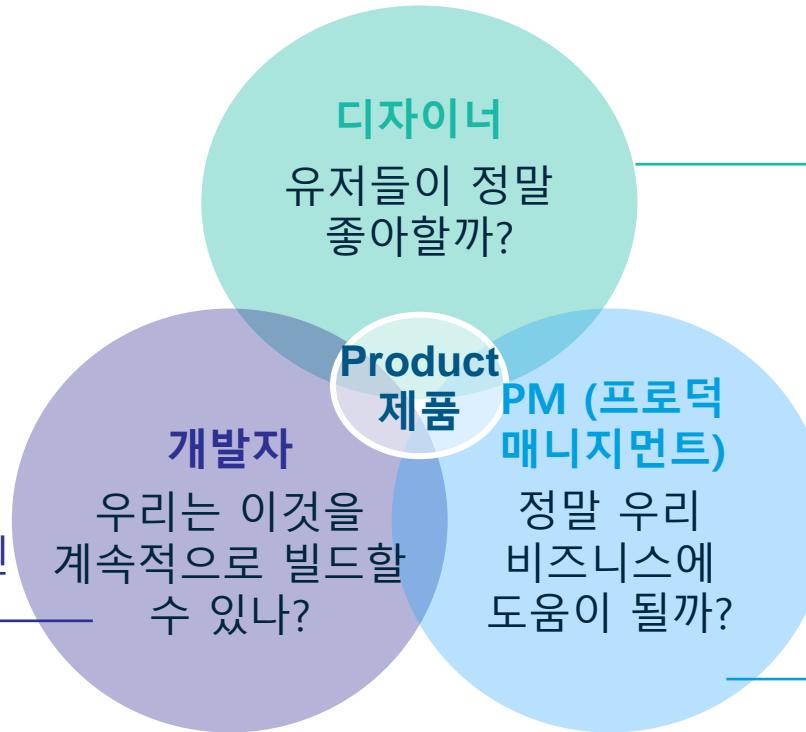
“하지만, 형식에 치우친 “애자일 따라하기” 보다는 제대로된 프로젝트의 수행을 통한 애자일 가치에 대한 진정한 “체득”과 “성공 경험”이 가장 중요함”

# 애자일 팀 구성 Three Roles

실현 가능성

이 프로젝트를 성공적으로  
수행하기 위해 필요한 기술적인  
복잡성은 무엇인가?

변화에 가장 능동적인 개발  
문화와 방법론을 만들기 위해서  
어떤 부분을 바꿔야 하는가?



고객을 위한 바람직한 방향성

고객이 진짜 가지고 있는 어려움이  
무엇인가? 우리가 맞게 개선하고  
있는가?

그 어려움을 어떻게 풀 수 있는가?

고객은 정말 이 애플리케이션을  
사랑할 것인가?

성공에 대한 기준

이러한 고객의 문제를 풀어줌으로서  
우리는 정말 가치있는 사업적인  
결과를 낼 수 있는가?

이런 것들을 어떻게 계속적으로  
측정할 것인가?

# 애자일의 하루와 철학



9.06am  
스탠드업 미팅

9:15- 12:30  
페어 제품 개발

12:30 – 13:30  
점심식사

13:30 – 18:00  
페어 제품 개발

18:00  
퇴근

## Pair (짝 활동)

- 하루종일 짝과함께 개발/디자인 진행
- 주기적으로 짝을 로테이션하여 지식 공유 촉진
- 짝 활동을 보조하는 소프트웨어 필요

## UCD (User Centric Design)

- 고객의 행동을 리드하는 것이 아닌, 고객의 행동에 맞춰가는 디자인 철학
- 애자일 팀에서 직접 엔드 유저 와의 인터뷰 진행

## TDD (Test Driven Dev)

- 제품 개발을 하기 전에 “무조건” 테스트 및 QA 부터 짜야 함
- 모든 테스트가 통과하고 언제나 동작하는 소프트웨어를 유지

# Timeline vs. Kanban Board Options

**Timeline Roadmap**

**vs.**

**Kanban Roadmap**



# Example Kanban Board for Product Roadmap

ProductPlan-3

Product Roadmap - Kanban

## Product Roadmap

	Backlog	To-Do	In Progress	Done
Web Team	<ul style="list-style-type: none"><li>3rd Party Integrations</li><li>New Admin Console</li><li>Security 2.0</li></ul>	<ul style="list-style-type: none"><li>On Premise Backup</li><li>Product Review</li><li>Self Service Portal</li></ul>	<ul style="list-style-type: none"><li>UI Improvements</li></ul>	<ul style="list-style-type: none"><li>Code Review</li><li>Shopping Cart Improvement</li></ul>
Mobile Team	<ul style="list-style-type: none"><li>Android Application</li><li>Mobile Mock Up</li><li>UX Improvements</li></ul>	<ul style="list-style-type: none"><li>Cloud Support</li><li>Interactive Dialogue Box</li><li>UX Improvements</li></ul>	<ul style="list-style-type: none"><li>Application Upgrade</li><li>Ticketing System</li></ul>	<ul style="list-style-type: none"><li>Q3 Priorities List</li><li>Ticketing System</li></ul>
Marketing Team	<ul style="list-style-type: none"><li>Customer Outreach</li><li>Lead Gen</li><li>Market Analysis</li></ul>	<ul style="list-style-type: none"><li>Pricing Review</li></ul>	<ul style="list-style-type: none"><li>SEO Plan</li></ul>	<ul style="list-style-type: none"><li>Content Review</li><li>Customer Outreach</li><li>Market Analysis</li><li>Performance Management</li><li>Proactive Email Campaign</li><li>SEO Plan</li></ul>

Drag and Drop coming soon

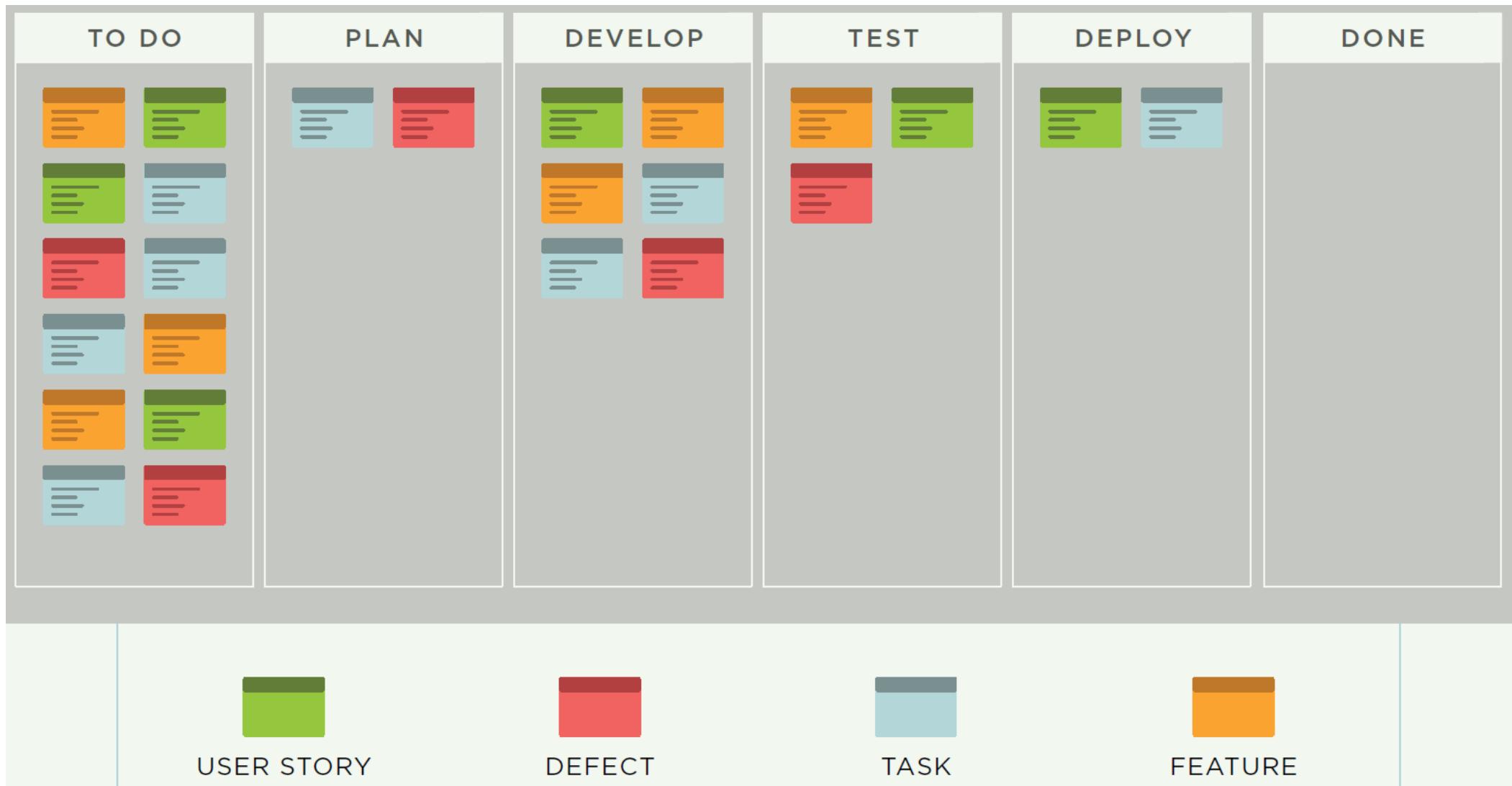
Settings Colors Filter Share Export History

Add Bar Add Container Add Lane

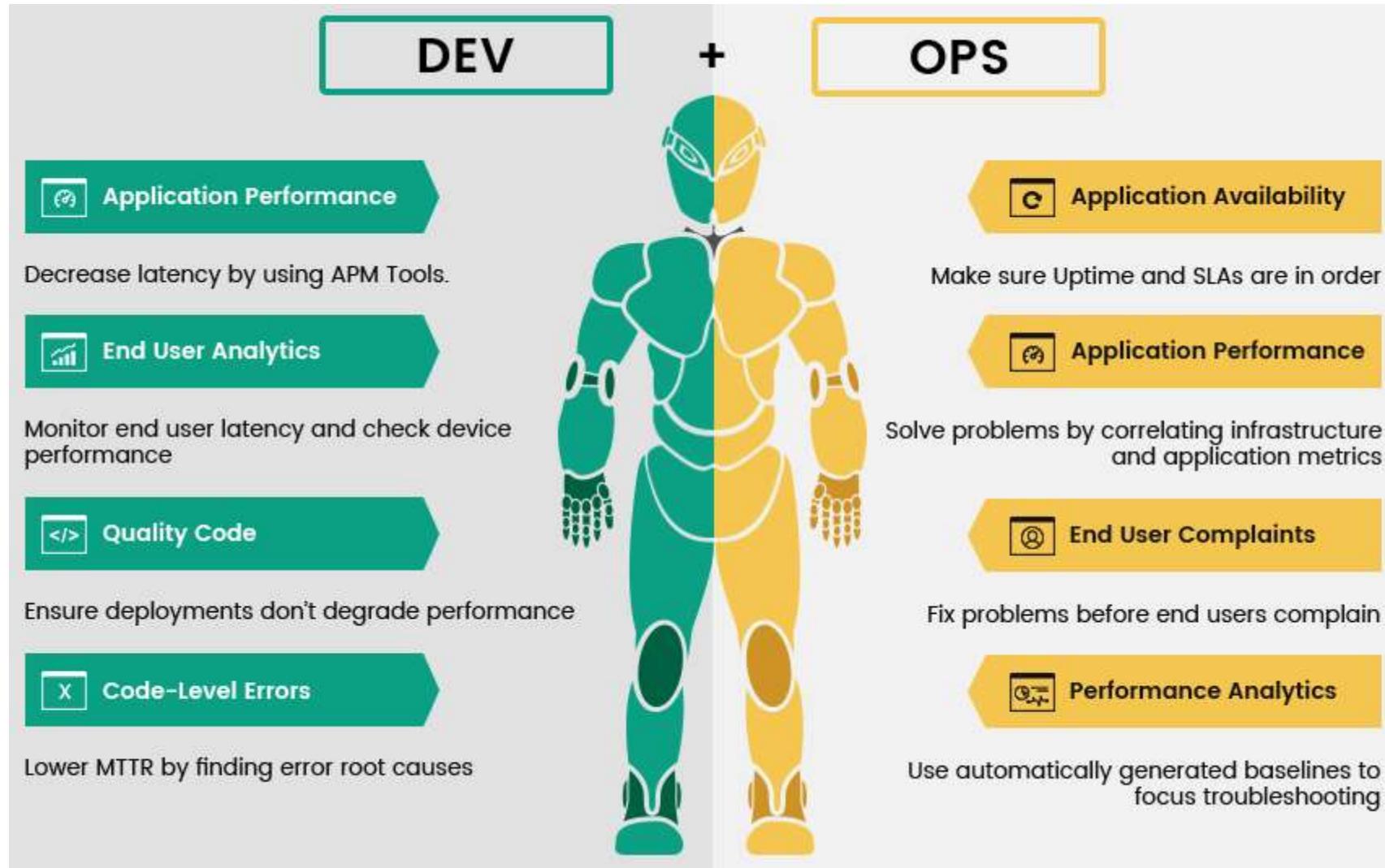
Strategic Goals

- Enhance Performance
- Internal Optimization
- Security Improvements

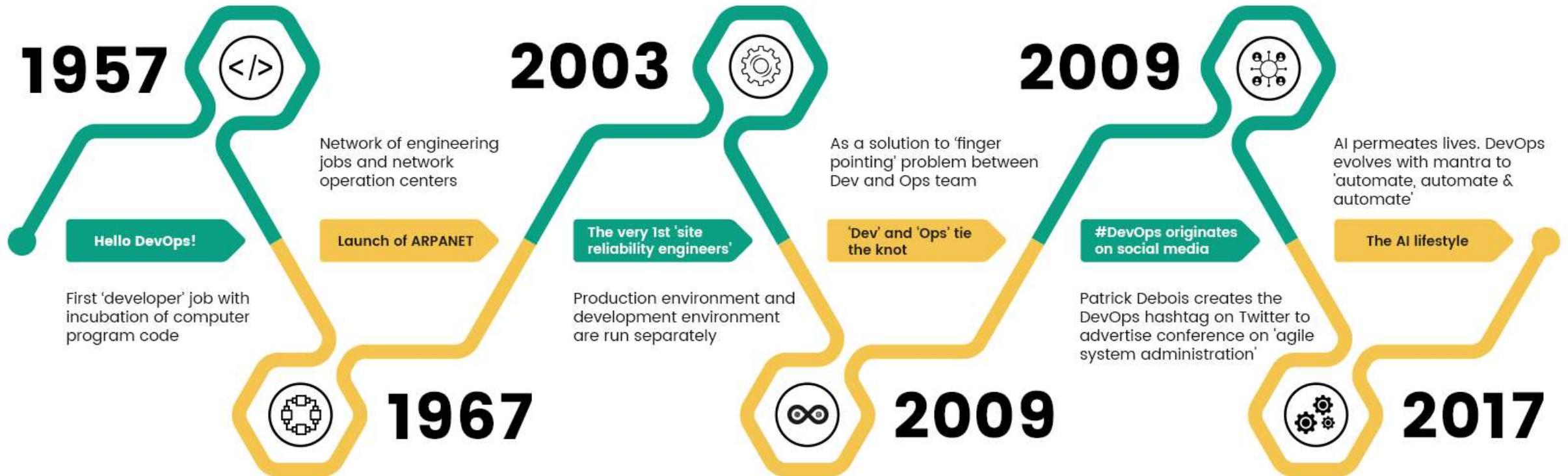
# Example Kanban Board for Agile DevOps



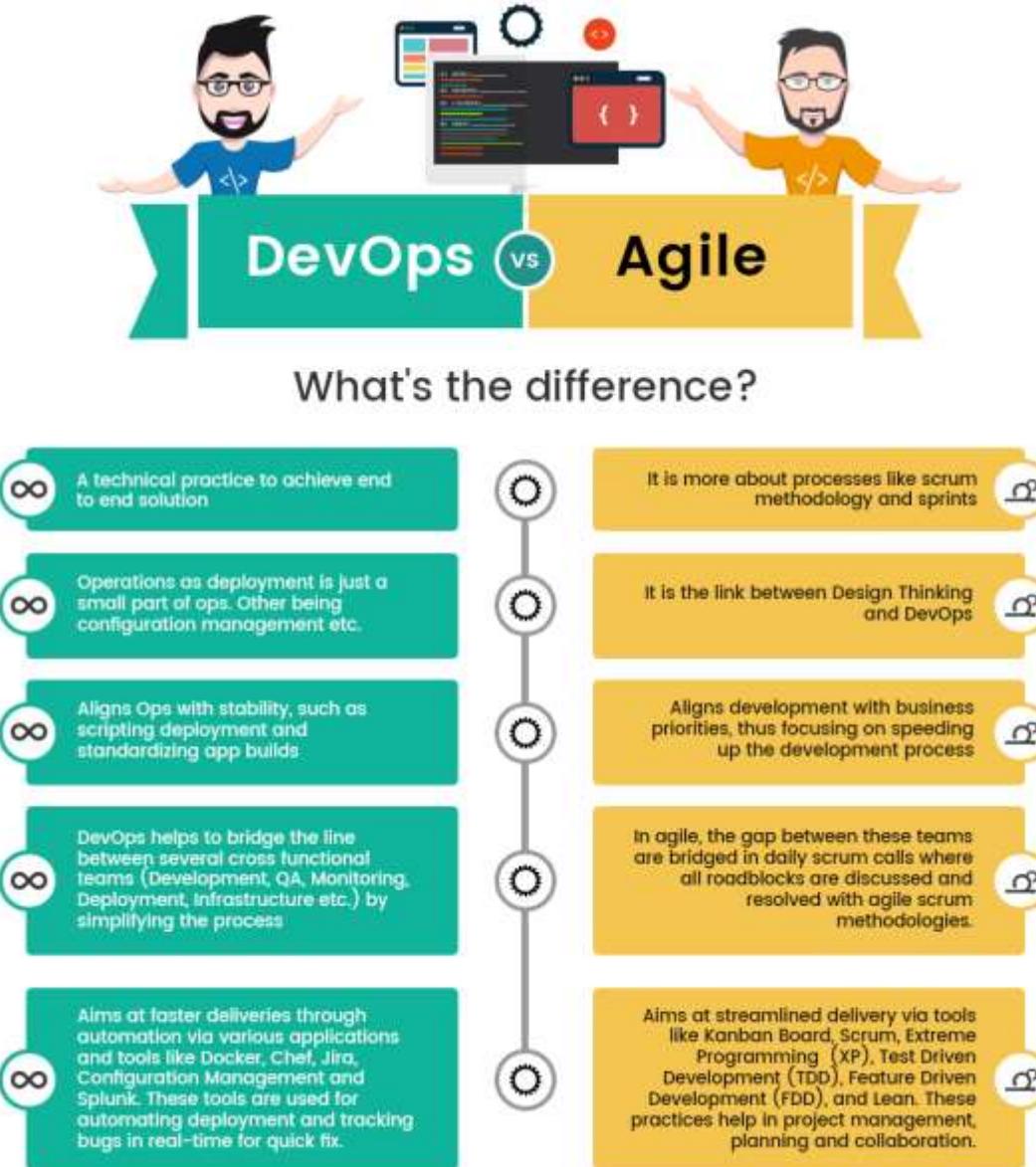
# DevOps



# History of DevOps



# DevOps & Agile



# **Part II**

# **MSA & CI/CD**

이 주문 웹페이지에서 마이크로서비스는 몇 개 일까요?

The screenshot shows the product page for the book 'Inviting Disaster: Lessons From the Edge of Technology' by James R. Chiles. The page includes the book cover, customer reviews, price information for Kindle, Hardcover, and Paperback, a detailed description of the book's content, and a 'Frequently Bought Together' section. On the right side, there is a sidebar with options to buy new or used, Prime delivery details, and other purchasing options like Add to Cart, Add to Wish List, and Sell on Amazon.

**Inviting Disaster: Lessons From the Edge of Technology** Paperback

by James R. Chiles (Author)

★★★★★ • 58 customer reviews

See all 7 formats and editions

Kindle \$10.23   Hardcover from \$0.01   Paperback \$12.06 ✓Prime

37 Used from \$0.01  
14 New from \$0.01  
90 Used from \$0.71  
18 New from \$8.16  
2 Collectible from \$8.00

Combining captivating storytelling with eye-opening findings, *Inviting Disaster* delves inside some of history's worst catastrophes in order to show how increasingly "smart" systems leave us wide open to human tragedy.

Weaving a dramatic narrative that explains how breakdowns in these systems result in such disasters as the chain reaction crash of the Air France Concorde to the meltdown at the Chernobyl Nuclear Power Station, Chiles vividly demonstrates how the battle between man and machine may be escalating beyond manageable limits — and why we all have a stake in its outcome.

Included in this edition is a special introduction providing a behind-the-scenes look at the World Trade Center catastrophe. Combining firsthand accounts of employees' escapes with an in-depth look at the

[Read more](#)

**Frequently Bought Together**

Price for all three: \$53.26

[Add all three to Cart](#) [Add all three to Wish List](#)

Show availability and shipping details

This item: Inviting Disaster: Lessons From the Edge of Technology by James R. Chiles Paperback \$12.06

The Logic Of Failure: Recognizing And Avoiding Error In Complex Situations by Dietrich Dorner Paperback \$12.36

Normal Accidents: Living with High-Risk Technologies by Charles Perrow Paperback \$28.82

Buy New ✓Prime \$12.06  
List Price: \$46.99  
Save: \$34.93 (74%)  
Only 16 left in stock (more on the way).  
Ships from and sold by Amazon.com  
Gift-wrap available.  
[Add to Cart](#)

Sign in to turn on 1-click ordering  
Want it tomorrow, May 27 Order within 2 hrs 22 mins and choose One-Day Shipping at checkout. Details

Buy Used ✓Prime \$9.22

Add to Wish List

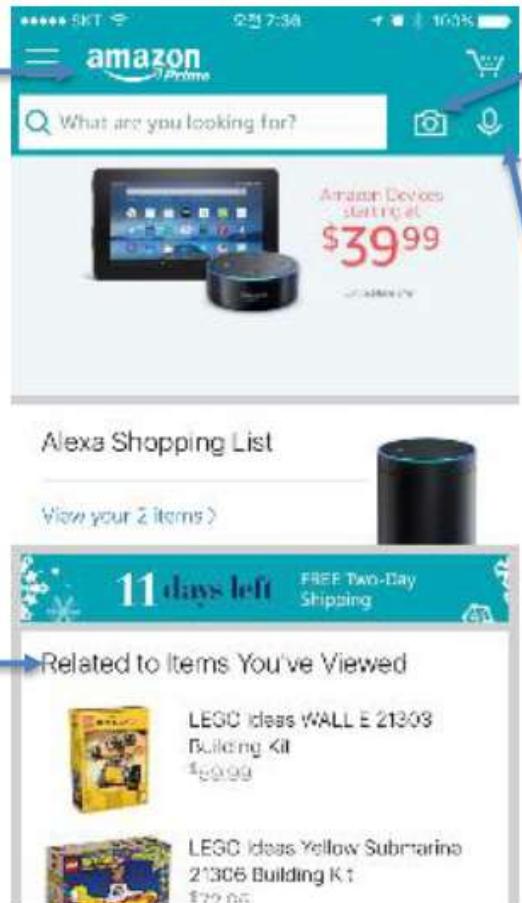
Have one to sell? Sell on Amazon

11개



# MSA in Amazon.com

아마존 프라임  
고객 차별화



이미지  
검색



최근 관심  
상품에 대한  
추천



음성인식  
제어



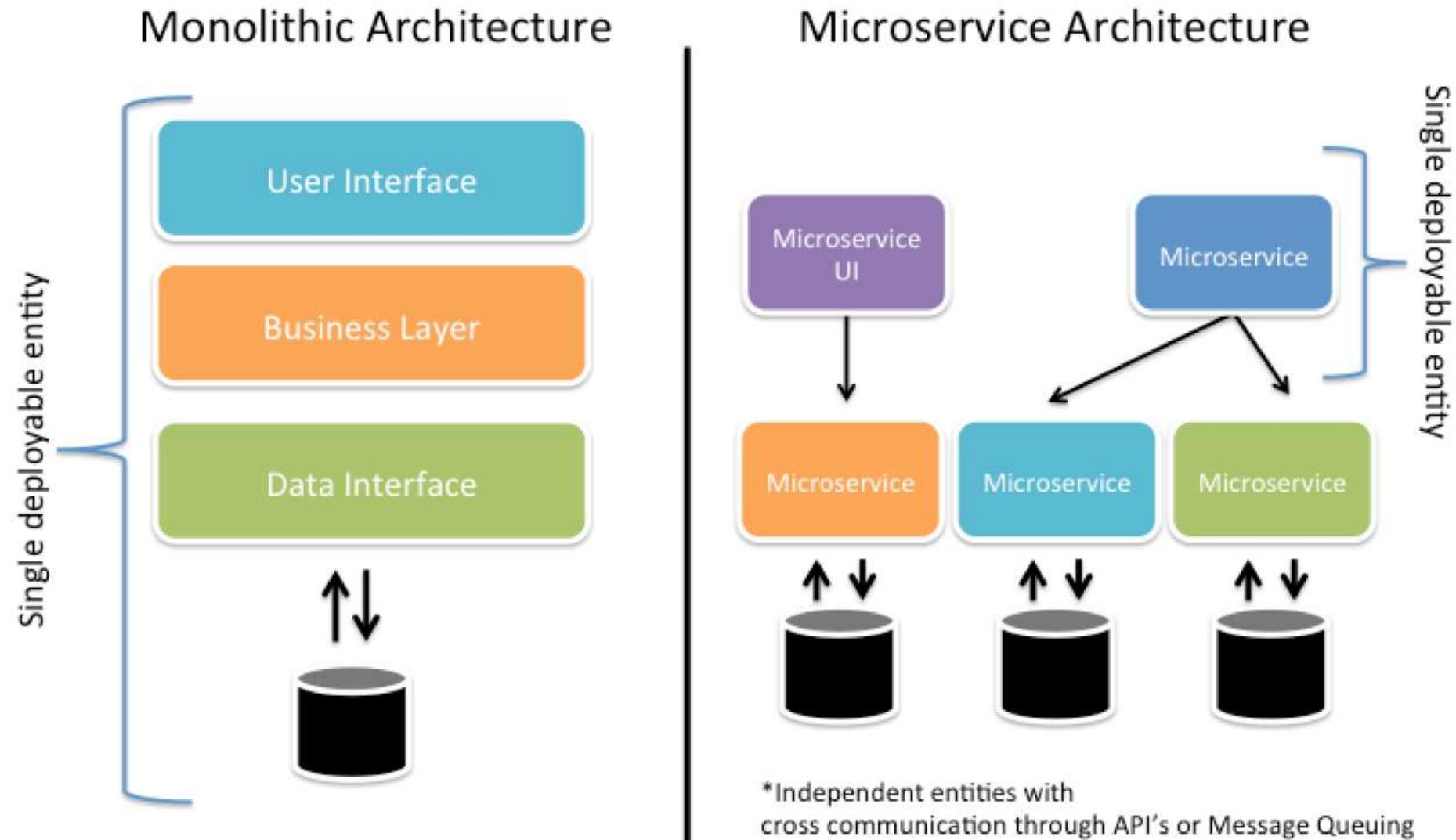
제품 브랜드  
이미지 태그



관련 상품  
리스트

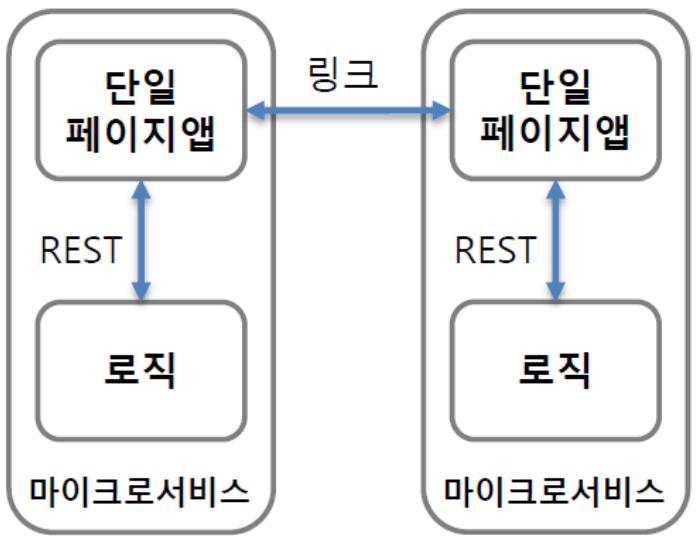
Where's my  
package?

내가 주문한  
상품에 대한  
상태 화면



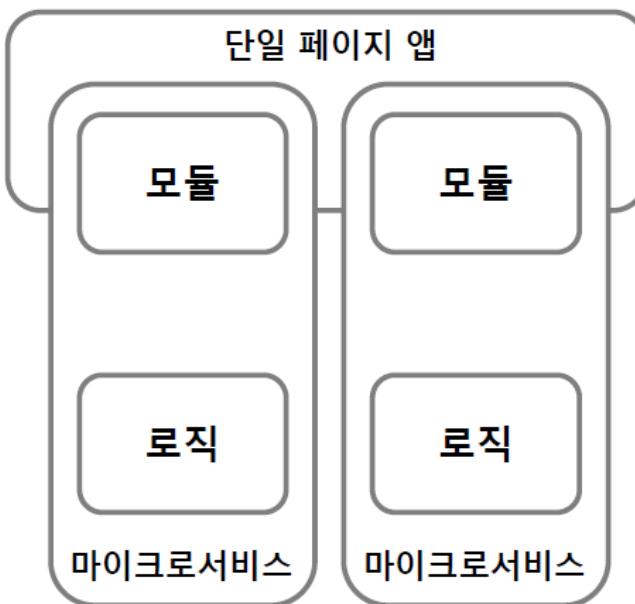
# MSA 유형

단일 페이지 앱을 갖는 마이크로서비스



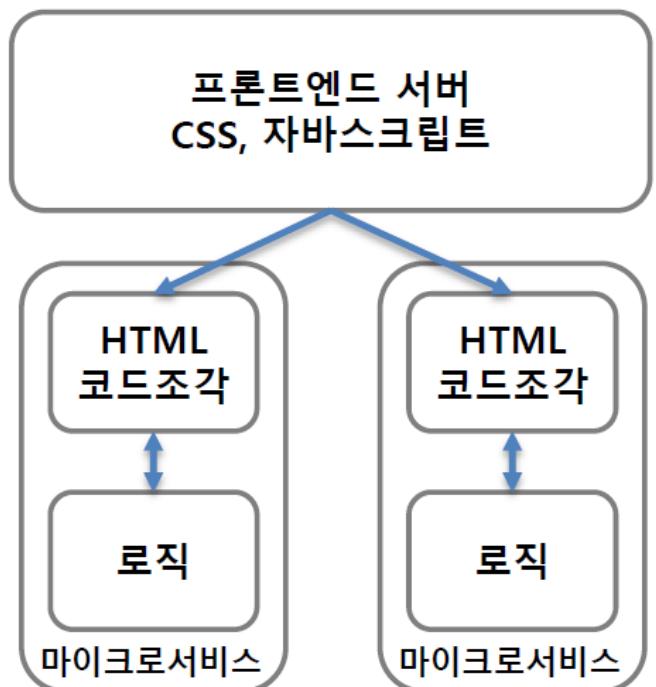
완전히 분리되며 독립적인  
서비스간의 연계를 위한 구성  
강력한 통합은 어려움

하나의 단일 페이지 앱을 공유하는  
마이크로서비스의 밀접한 통합



마이크로서비스의 사용자  
인터페이스에 대한 강력한  
통합이 필요한 경우

프론트엔드 서버를 이용한 통합



강력한 통합을 위한 대안  
서로 독립적으로 생산환경에  
적용 가능(예, 자바포틀릿)

An approach that helps reduce the risk of delivering changes by allowing for more frequent and incremental updates to applications in production

## Continuous integration (CI)

The practice of regularly merging the work products of individual developers together into a repository. The primary purpose is to enable early detection of integration bugs, resulting in tighter cohesion and more development collaboration.

## Continuous delivery (CD)

A software engineering approach in which teams produce software in short cycles and use processes ensuring that the software can be reliably released to production at any time.

# CI/CD Pipelines



## Build

*The codebase is checked out to a specific version and artifacts (e.g., Docker containers) are built.*



## Deploy

*Artifacts are deployed into an environment.*



## Test

*Tests (unit, integration, vulnerability, performance, etc.) are performed to ensure application quality.*



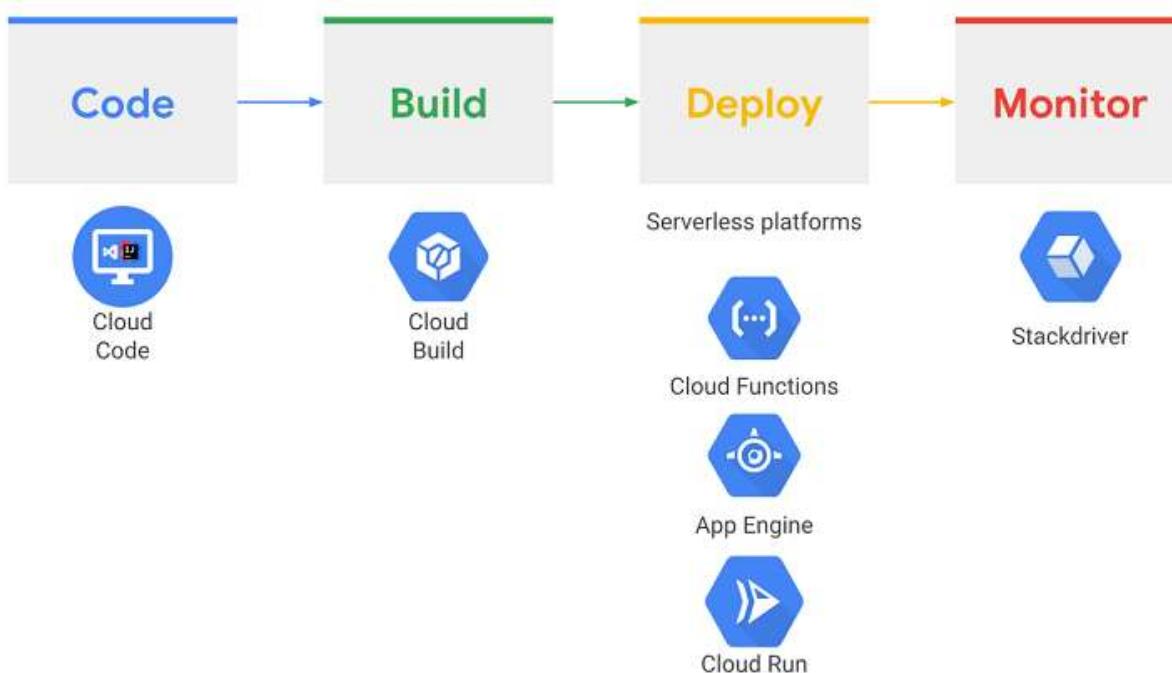
## Approve

*User determines whether a pipeline should proceed to the next stage.*

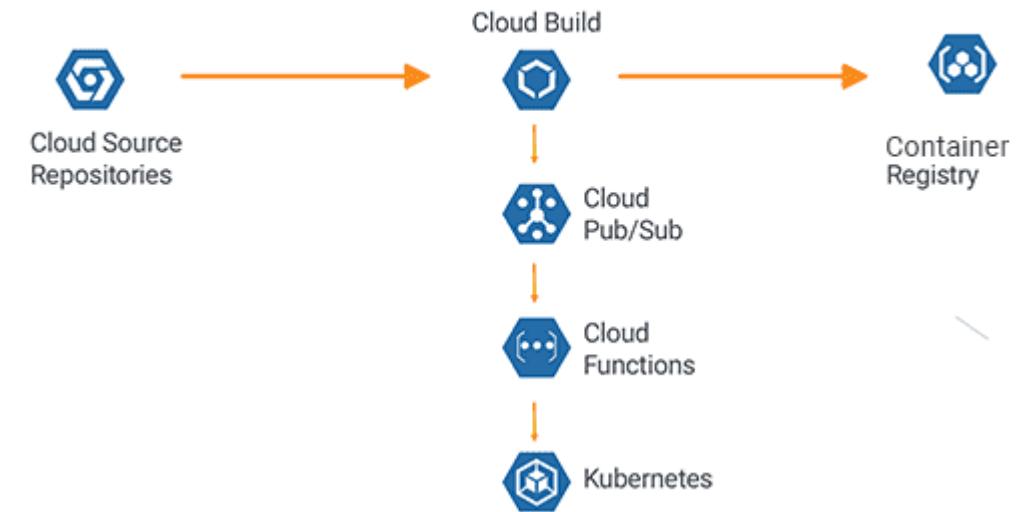
A **pipeline** is a process that takes a version of an application codebase and performs all steps necessary to release it to production. Pipelines can be triggered manually, or they can be triggered automatically when changes are pushed to the codebase. They are generally composed of separate stages, most commonly these four.

# Google GCP CI/CD Example

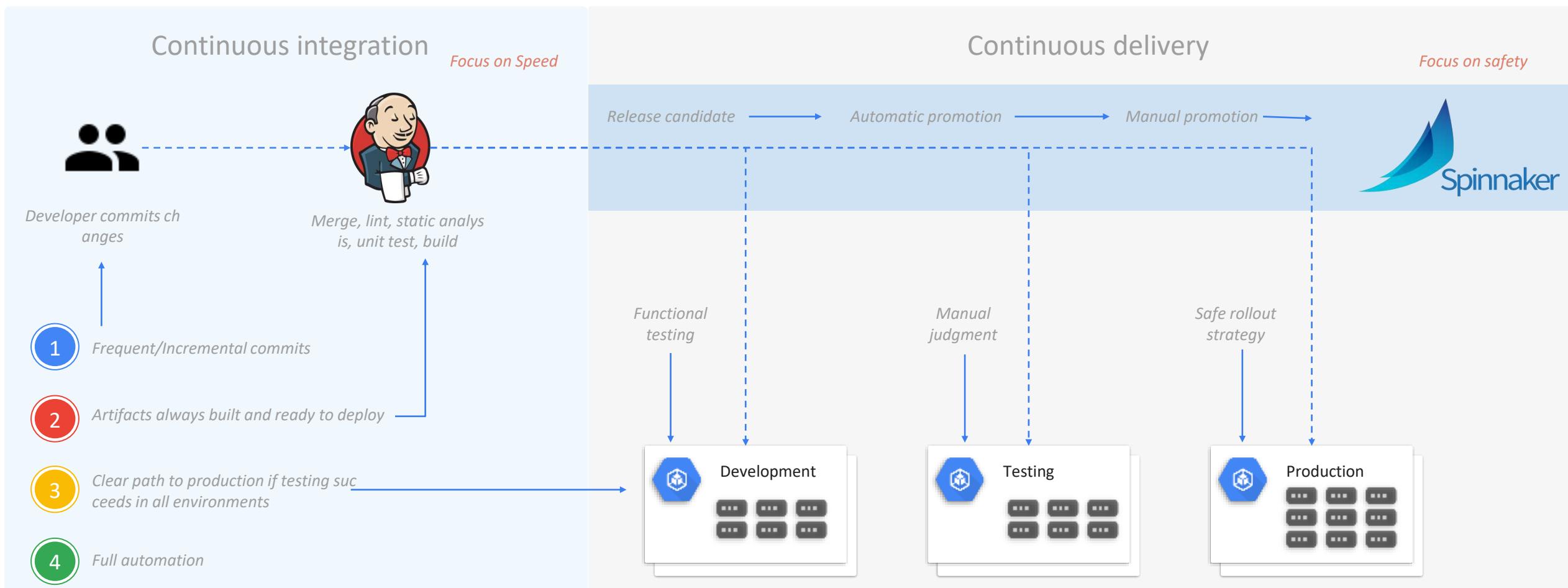
- CI/CD DevOps Pipeline



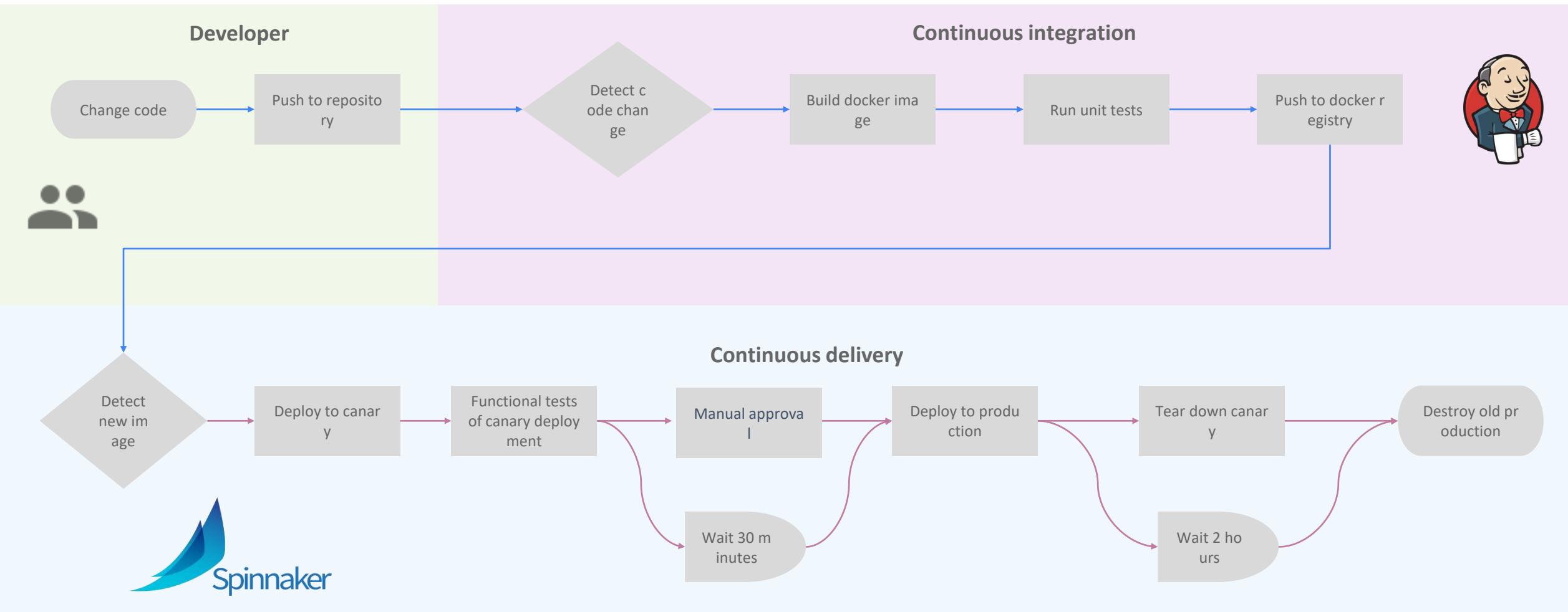
## DevOps Pipeline on Google Cloud Platform



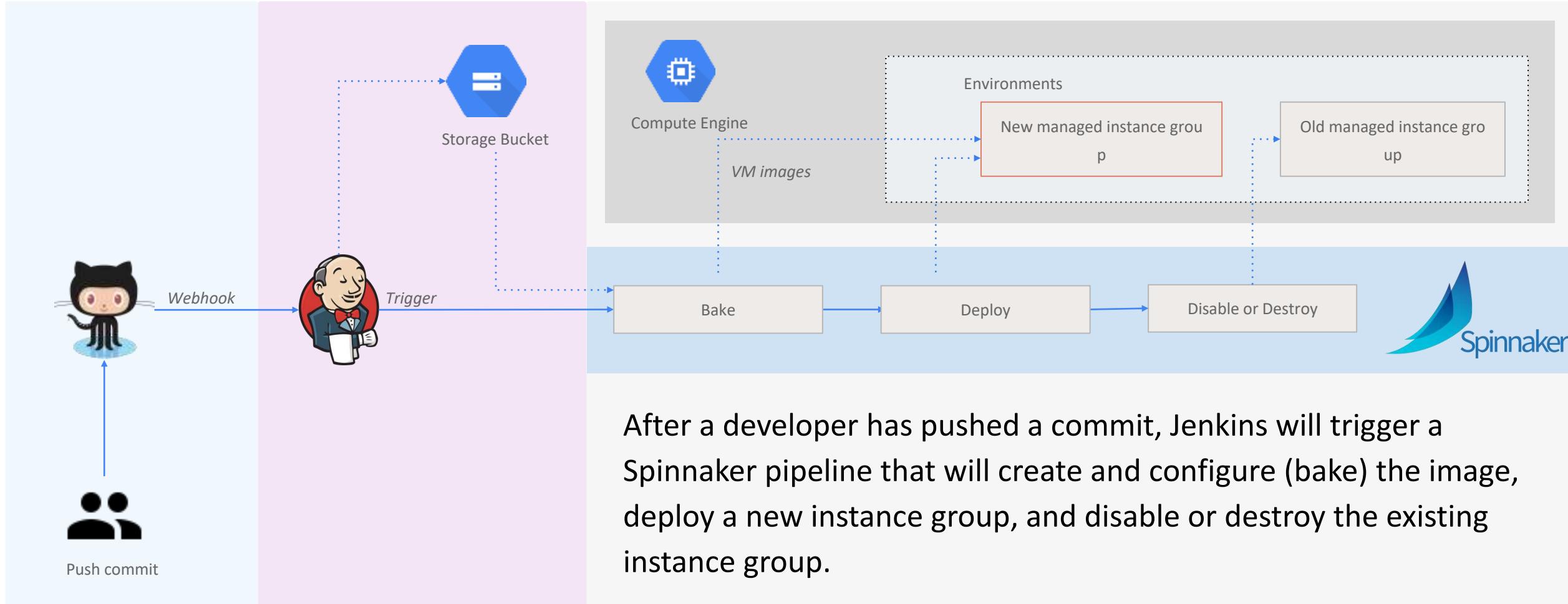
# Example CI/CD Pipelines



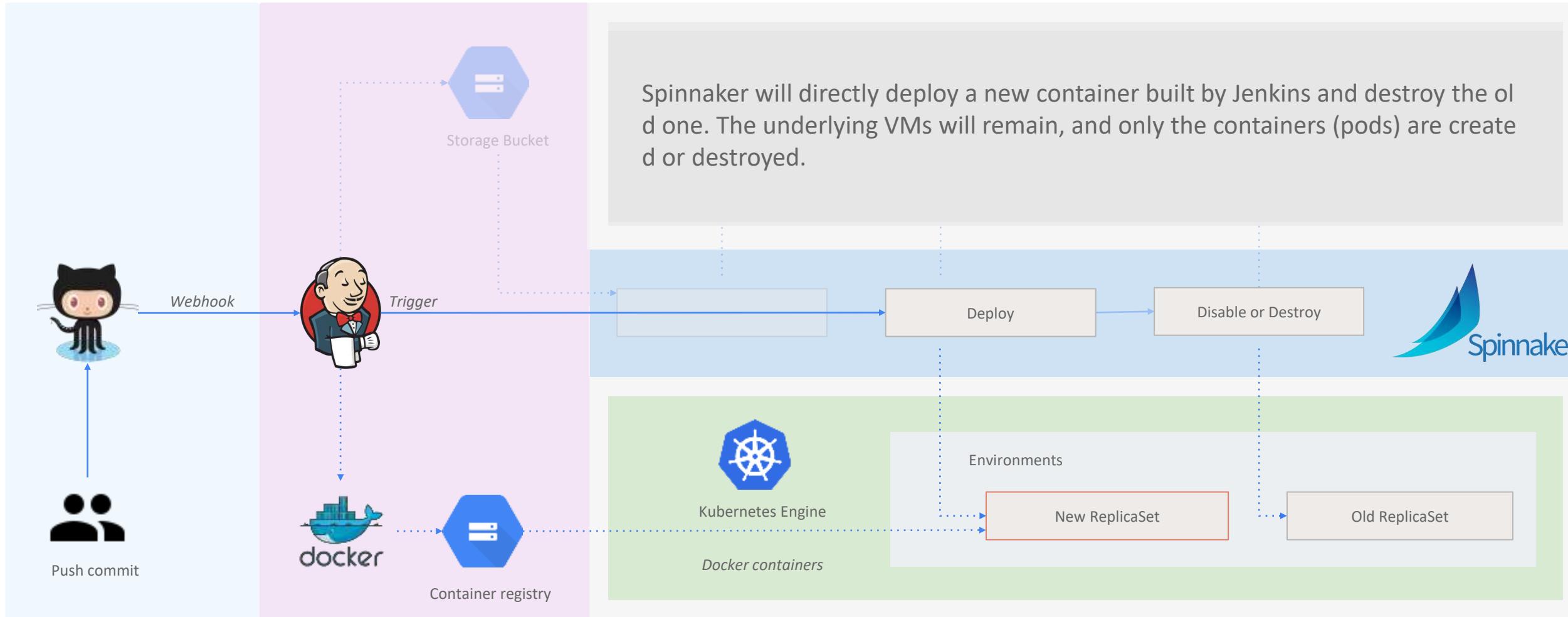
# Pipelines Process Flow



# Continuous Deployment to VMs



# Continuous Deployment to Containers



# Best practices for effective pipelines

## Quality

- Changes are thoroughly tested before being deployed to production.
- Production deploys are performed in a safe manner (e.g., canary or blue/green to reduce risk), with failures triggering an automatic rollback to the prior version.

## Speed

- Pipelines are fully automated with the exception of manual-approval gates.
- No bottlenecks (i.e., delays in promoting a given code change do not block other, potentially more critical changes from being promoted).

## Visibility

- Pipelines are fully integrated with the SCM branching and merging strategy.
- Important events (failures, etc.) trigger alerts to the appropriate stakeholders.
- Each artifact is marked with the metadata necessary to trace how it was produced and the path it has taken in the pipeline.

## Scalability

- Stages that are common across pipelines are organized into a library to be centrally managed and highly reused.

# Key decisions

- 1** How should pipelines be triggered?
- 2** What tests need to be run?
- 3** What environments need to be deployed?
- 4** What strategies should be used for production deploys?
- 5** Under what conditions should notifications be sent and to whom?
- 6** At which points should approvals be required to proceed?

## Module 7

# Jenkins를 이용한 CI & CD 이해와 실습

# **Part I**

## **CI/CD Tools - Jenkins**

# Example marketplace CI/CD tools

## Jenkins



Jenkins is one of the earliest open-source continuous integration servers and remains the most common option in use today.

## GitLab CI



GitLab CI is a continuous integration tool built into GitLab, a git repository hosting and development tools platform.



## Spinnaker

Spinnaker is an open-source, multicloud, continuous delivery platform that helps you release software changes with high velocity and confidence.



## Drone

Drone is a modern CI/CD platform built with a container-first architecture. Running builds with Docker, a container-based workflow, is at the core of Drone's design.



## CircleCI

The CircleCI continuous integration and delivery platform makes it easy for teams of all sizes to rapidly build and release quality software at scale.

# Jenkins software — continuous integration



Jenkins is an open-source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat. It supports version-control tools, including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, ClearCase, and RTC, and can execute Apache Ant, Apache Maven, and sbt-based projects, as well as arbitrary shell scripts and Windows batch commands.

# Continuous integration: Jenkins tool review

## Strengths

- Mature product, extensive community support
- Simple architecture
- Entirely pipeline-as-code (Jenkinsfile), UI is just a view
- Deep level of control in pipelines (groovy-based)
- Pipeline libraries for sharing and reuse
- Slaves created as pods during job run, offering efficient resource usage

## Weaknesses

- Lacks integration with Kubernetes objects, often supplemented with tools like Helm
- HA options not natively supported

# Spinnaker — continuous Deployment



Spinnaker is an open-source, multi-cloud continuous delivery platform for releasing software changes with high velocity and confidence.

# Continuous deployment: Spinnaker tool review

## Strengths

- Tight integration with Kubernetes workload resources (Pod, ReplicaSet, Service, Ingress, etc.)
- Agnostic on deployment destination (Kubernetes Engine, Google Compute Engine, AWS EC2, etc.)
- Active contributions by Netflix, Google, CoreOS, and others
- HA options available

## Weaknesses

- Fairly new product that lacks a high level of community support
- Primarily for CD, not CI — often still needs Jenkins or another CI tool
- Relatively complex architecture
- Currently, largely UI based

# Supporting tools

## Source code repository

A file archive facility where source code is kept and used by multi-developer projects to handle various versions. Products include GitHub, Bitbucket, and Google Cloud Source Repositories.

## Container registry

A storage and content delivery system, holding named Docker images, available in different tagged versions. Products include Quay, Docker Hub, Nexus, and Google Container Registry.

# Key decisions

- 1** Which tools and supporting tools should be used to implement CI/CD?
- 2** How will user access to these tools be managed?
- 3** How will pipelines be managed and reused across applications?

# **Part II**

# **Example CI/CD Architectures**

# **On-Premise CI/CD**

# Before CI/CD

Developer runs all tests, creates all artifacts, and generates all documentation on local machine.

- 1 Run local tests and push code to code repository
- 2 Push deployment artifact to prod

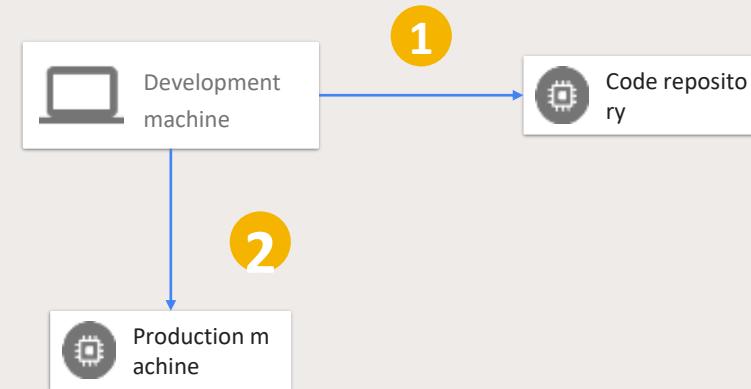
## Advantages

- Simple installation
- Simple deployment of code
- Best for single contributor projects
- Secured within private datacenter

## Disadvantages

- Slow deployments
- Error prone
- No fail-overs
- Geographically limited to datacenter

On-premises network



# On-premises CI/CD

Developer environment, infrastructure (code repository and CI/CD pipeline), and production servers as a CI/CD solution

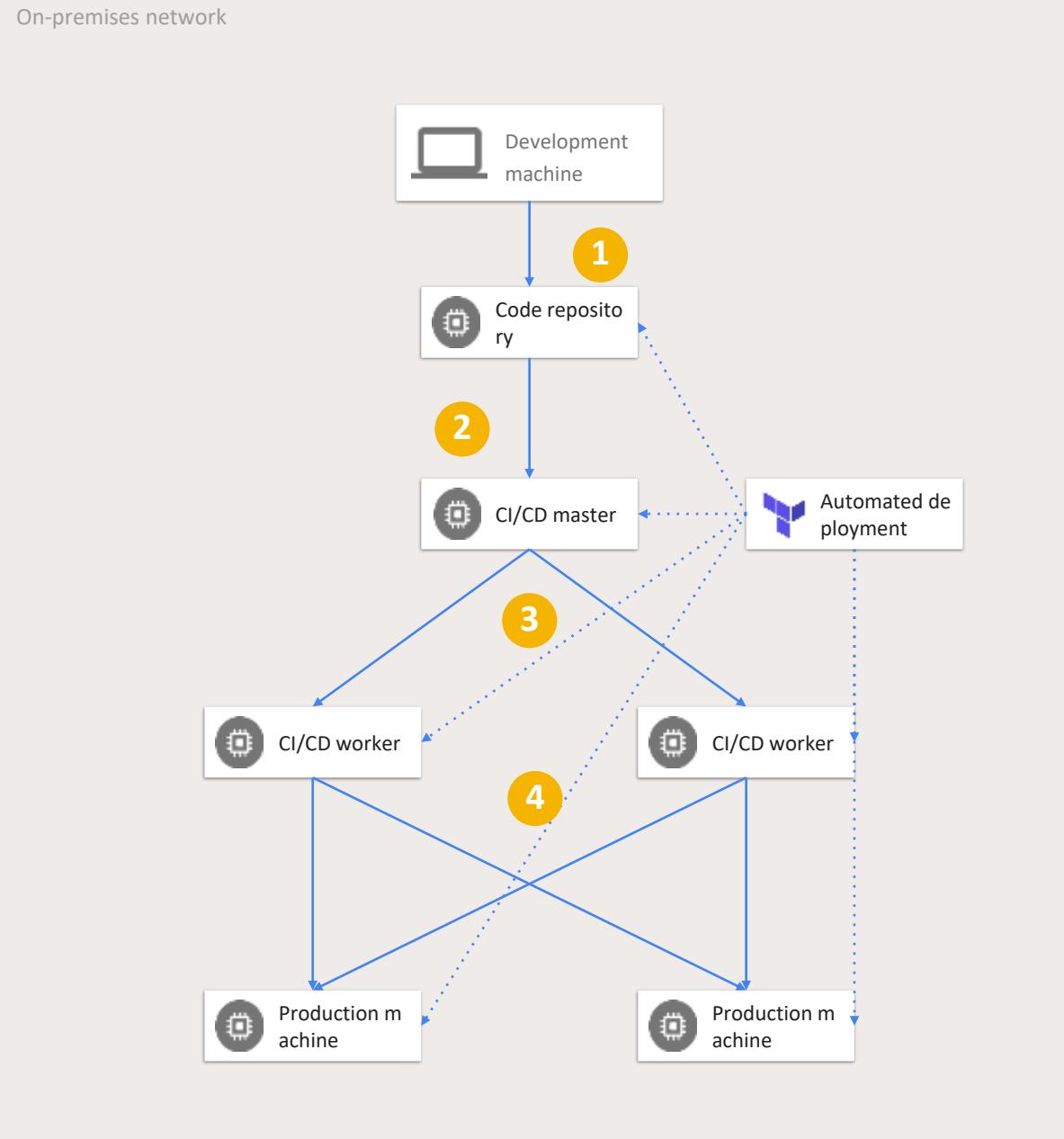
- 1 Push code to code repository
- 2 Repo notifies CI/CD master
- 3 CI/CD master delegates work to CI/CD workers
- 4 CI/CD workers successfully test build and deploy to prod

## Advantages

- Fast deployments
- Automated infrastructure
- Secured within private datacenter
- Integrated testing

## Disadvantages

- Single point of failure
- No automatic scaling
- Geographically limited to datacenter



# On-premises CI/CD with Kubernetes

Infrastructure cluster contains pods for CI/CD, and deployment artifacts run inside of the nodes in the production cluster.

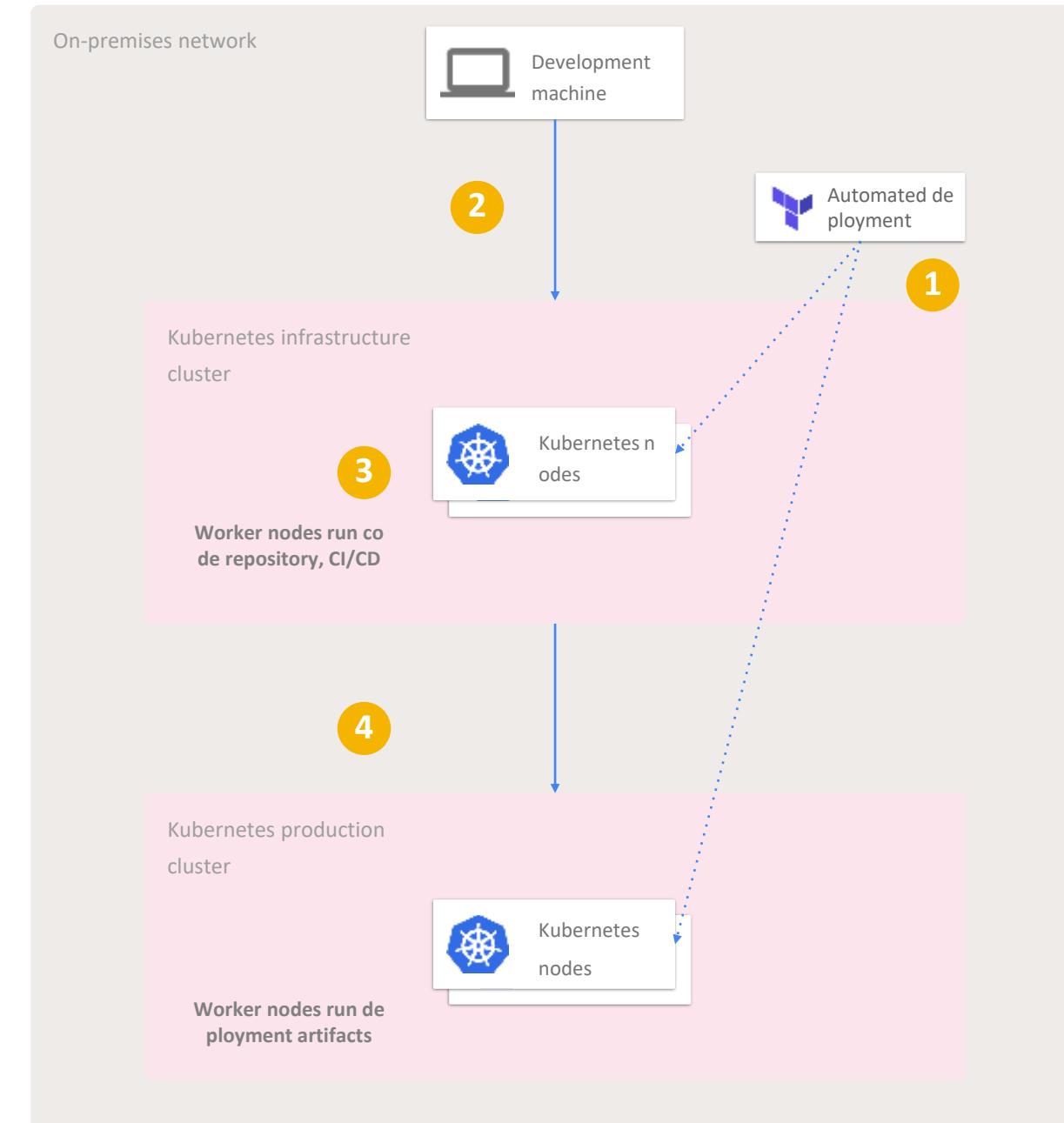
- 1 Terraform provisions infrastructure
- 2 Developer pushes code
- 3 CI/CD master delegates work to CI/CD workers
- 4 CI/CD workers successfully test build and deploy to prod

## Advantages

- Fast deployments
- No single point of failure
- Nginx load balancing

## Disadvantages

- Geographically limited to datacenter
- No automatic scaling



# On-premises CI/CD with Kubernetes

Load balancers located in front of the infrastructure cluster and production cluster. Two of each cluster exist for failovers.

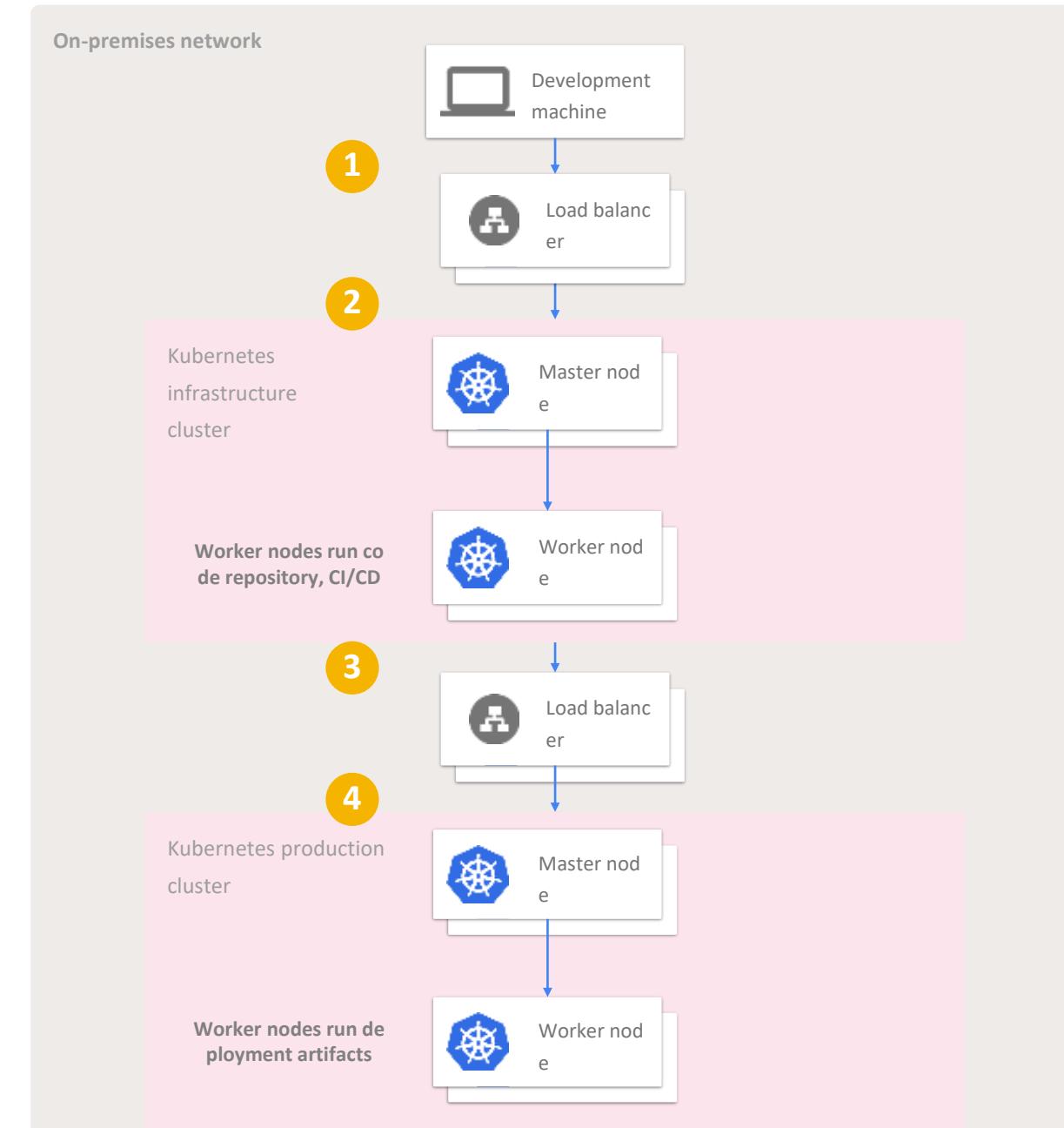
- 1 Git push connect to load balancers
- 2 Load balancer connects master node to delegate work to worker nodes
- 3 Worker nodes delegate work to master node via second load balancer
- 4 Master node delegates tasks to worker nodes, which successfully test build and deploy

## Advantages

- Automatic scaling
- No single point of failure
- Nginx load balancing

## Disadvantages

- Additional resources required
- Geographically limited to datacenter



# On-premises CI/CD with Kubernetes

git and Jenkins with minions are located in clusters in order to provide high availability.

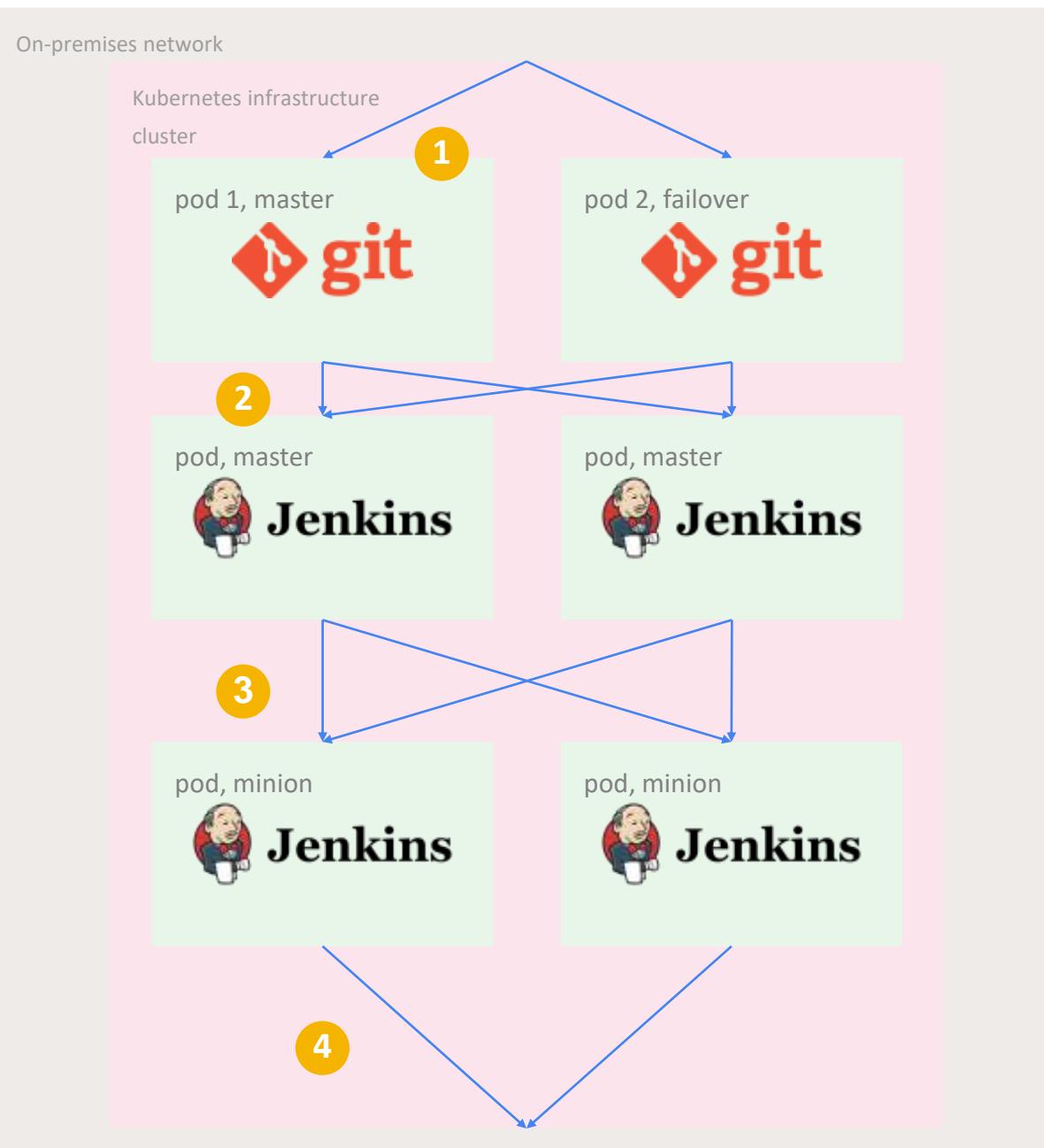
- 1 Load balancer connects to master node to access git service running on pod 1
- 2 Repo notifies Jenkins master of new code
- 3 Jenkins master commands Jenkins minions to test build
- 4 Workers successfully test build and deploy to prod

## Advantages

- Automatic scaling
- No single point of failure
- Nginx load balancing

## Disadvantages

- Additional resources required
- Geographically limited to datacenter



# Cloud CI/CD

# GCP CI/CD deployments

An example CI/CD deployment strategy on Google Cloud Platform products

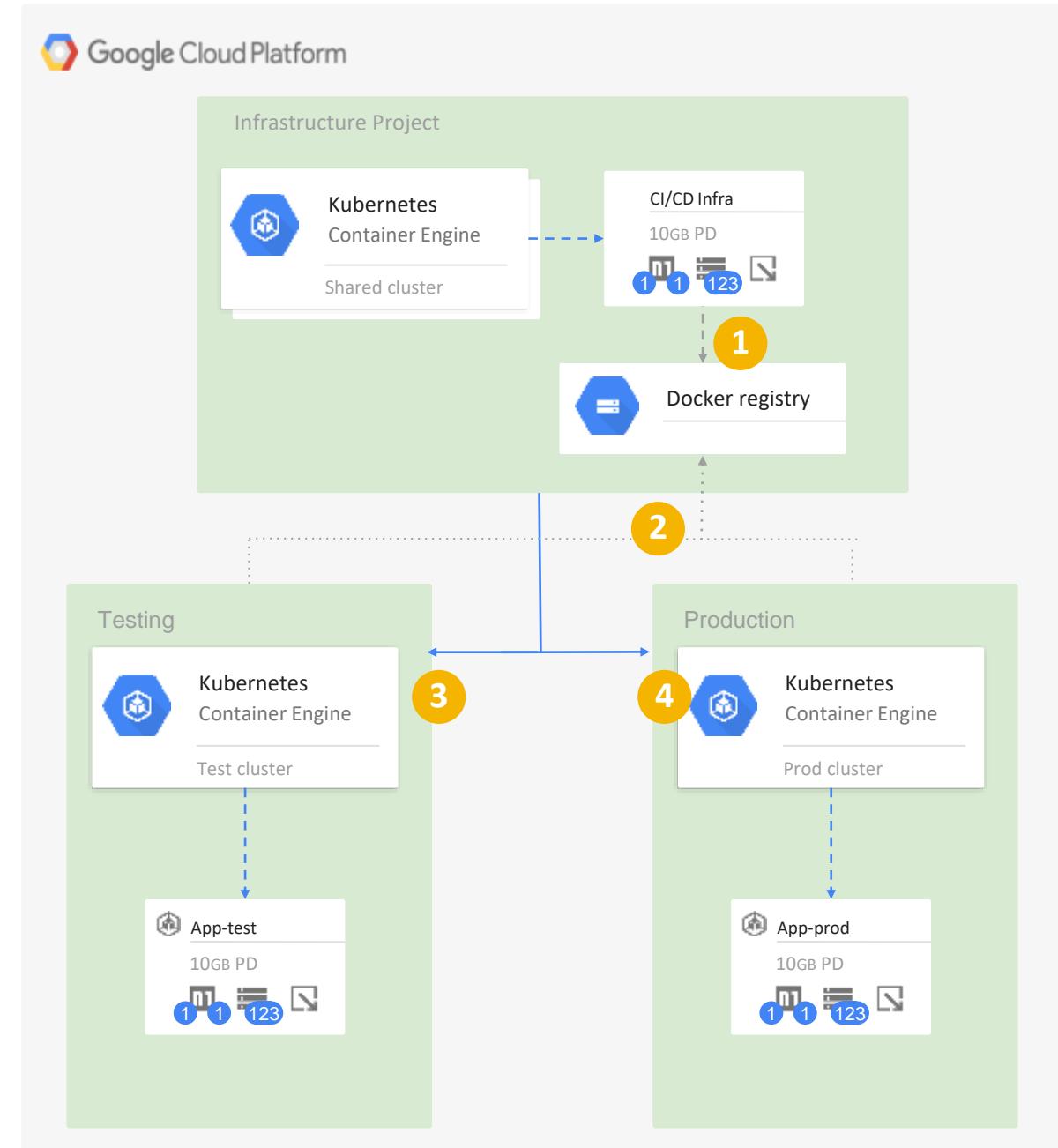
- 1 Push to Docker registry
- 2 Pull from docker registry
- 3 Deploy to testing environment
- 4 Deploy to production environment

## Advantages

- Fast deployments
- No single point of failure

## Disadvantages

- Additional resource usage



# Cloud push CI/CD with Kubernetes

**Node details:** Here, we have the exact same setup with the only exception being that the end resulting artifact is pushed into Kubernetes running on Google Compute Engine instances. Also note that Terraform can still deploy these cloud instances.

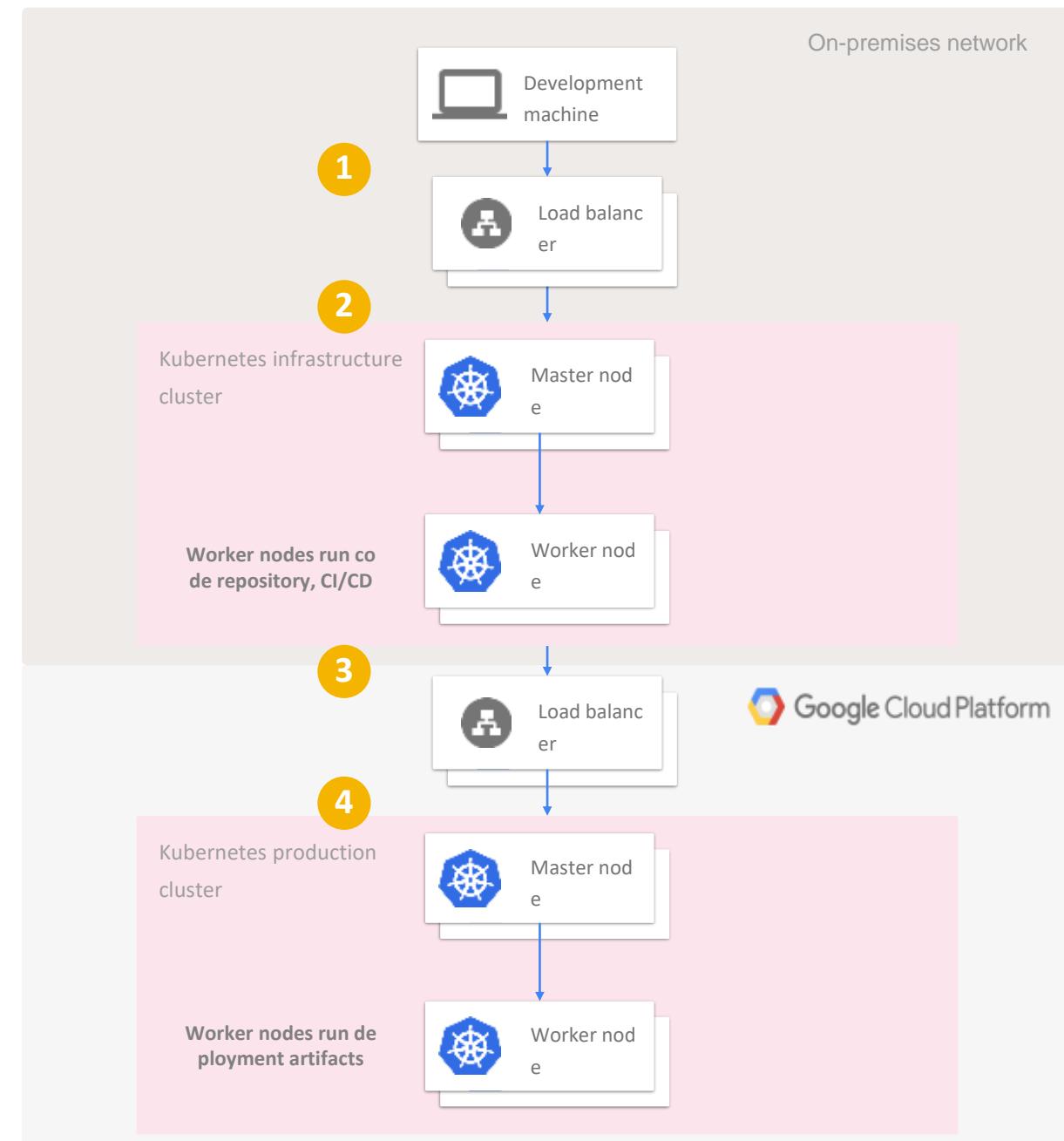
- 1 *Git push to load balancers*
- 2 *Load balancer connects master node to delegate work to worker nodes*
- 3 *Worker nodes delegate work to master node via second load balancer*
- 4 *Master node delegates tasks to worker nodes, which successfully test build and deploy*

## Advantage

- **S** Fast deployments
- No single point of failure

## Disadvantages

- Additional resource usage



# Cloud push CI/CD with Kubernetes

Here, we have a similar setup to the Cloud Push scenario, but here we develop in the cloud and then on-premises “pulls” the data in.

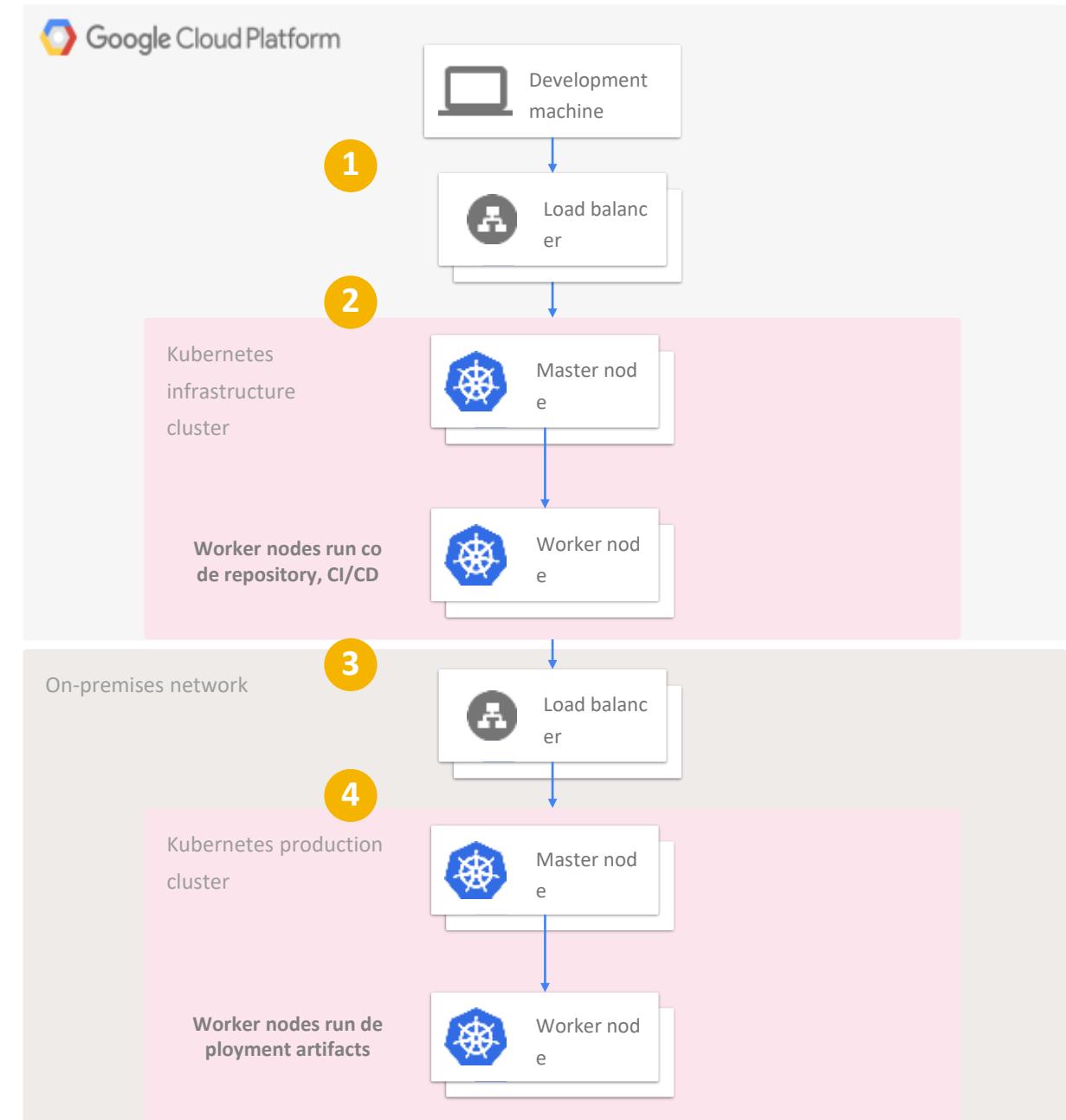
- 1 *Git push to load balancers*
- 2 *Load balancer connects master node to delegate work to worker nodes*
- 3 *Worker nodes delegate work to master node via second load balancer*
- 4 *Master node delegates tasks to worker nodes, which successfully test build and deploy*

## Advantages

- Fast deployments
- No single point of failure

## Disadvantages

- Additional resource usage
- Latency concerns connecting back to on-premises



# **Hybrid CI/CD**

# Open hybrid CI/CD with Kubernetes, unified network

Similar to on-premises architecture, but load balancers run on Google Compute Engine instances and each network has a set of K8s nodes. Container Registry also holds the Docker image containing the deployment artifact and prod has CI/CD pods.

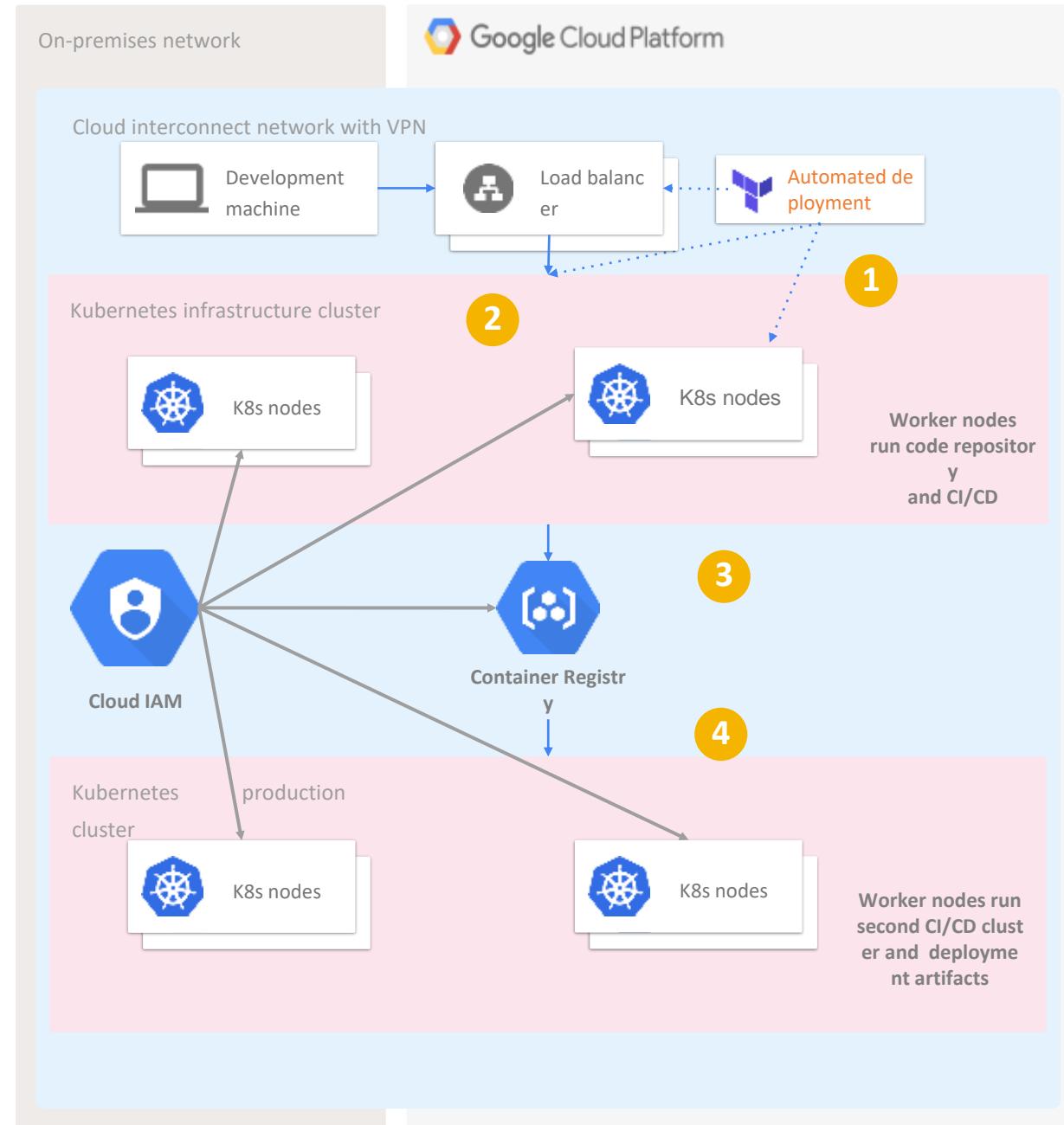
- 1 Terraform provisions necessary resources
- 2 Developers push code to code repo via load balancer
- 3 Pushes the Golden Image DA to the Container Registry
- 4 CI/CD pods in prod deploy the new image to prod

## Advantages

- Centralized auth in Cloud IAM
- Redundancy allows for more uptime

## Disadvantages

- Latency from VPN connection
- Additional egress costs for out-of-zone access



# Open hybrid CI/CD with Kubernetes, unified network

As before, but K8s load balancers run on Google Compute Engine instances, and one master cluster and worker cluster run on-premises and another two clusters run on GCP in Compute Engine instances. Cloud portion can scale up as needed, has minimal administrative overhead and enables high availability for your customers and developers.

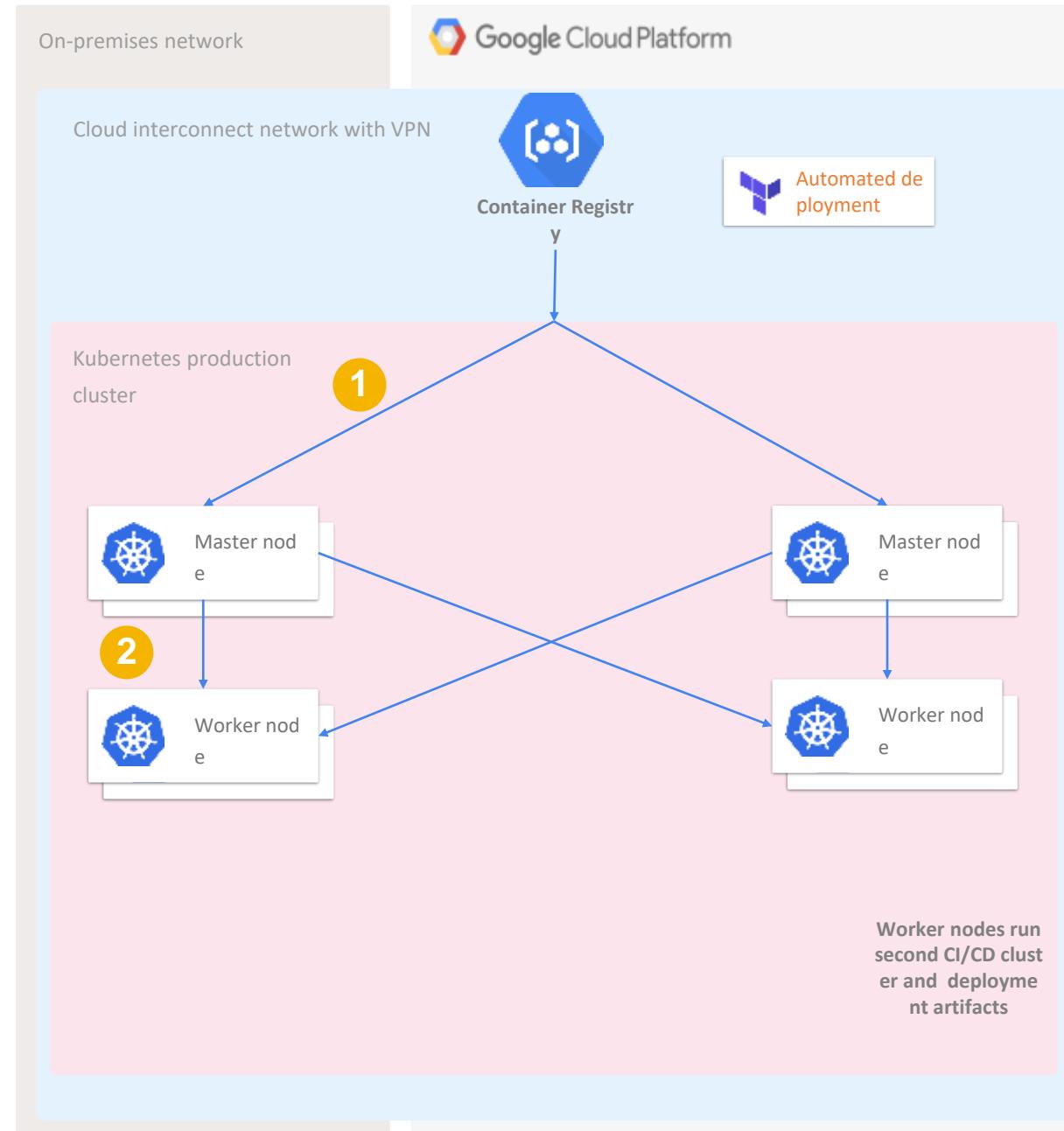
- 1 *The Container Registry pushes a notification of a new deployment image to the CI/CD tool running in the cluster*
- 2 *The instances then deploy the new image to pods in production*

## Advantages

- Fast deployments
- No single point of failure
- Lightweight

## Disadvantages

- Additional resource usage



# Open hybrid CI/CD with Kubernetes, unified network

Here, we can see a minimal Jenkins HA cluster (see next slide) pull a new Docker golden image from the Container Registry and deploy it to the same production cluster. Note that with a couple more production pods with the golden image you could have a rolling blue-green deployment strategy.

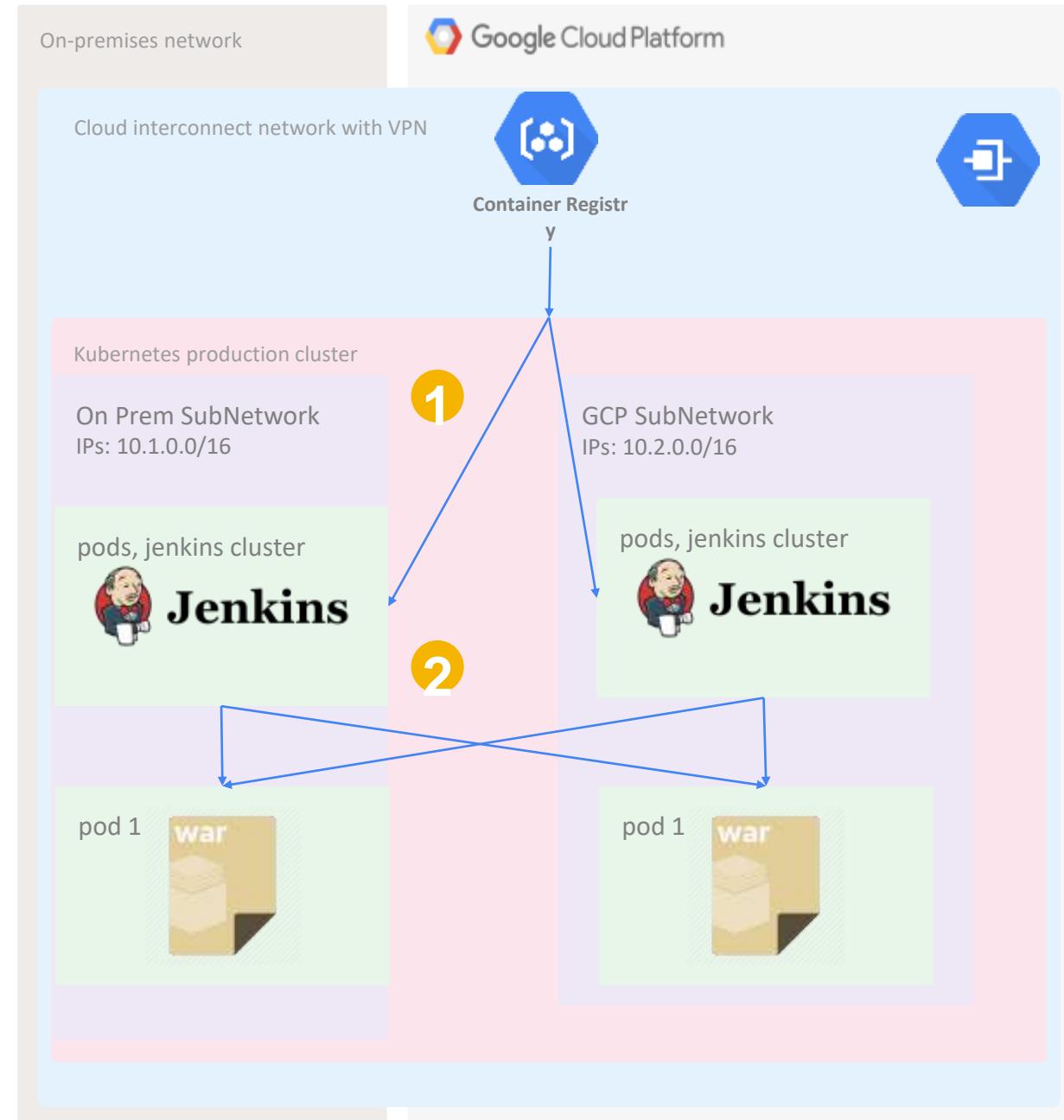
- 1 Jenkins pulls a new golden image
- 2 Jenkins deploys golden image

## Advantage

- S Fast deployments
- No single point of failure

## Disadvantages

- Additional resource usage
- Latency concerns connecting back to on-premises



# Open hybrid CI/CD with Kubernetes, separate networks

K8s load balancers run on Google Compute Engine instances, while one master cluster and worker cluster run on-premises and another two clusters run on GCP in Compute Engine instances. Cloud portion can scale up as needed, has minimal administrative overhead, and enables high availability.

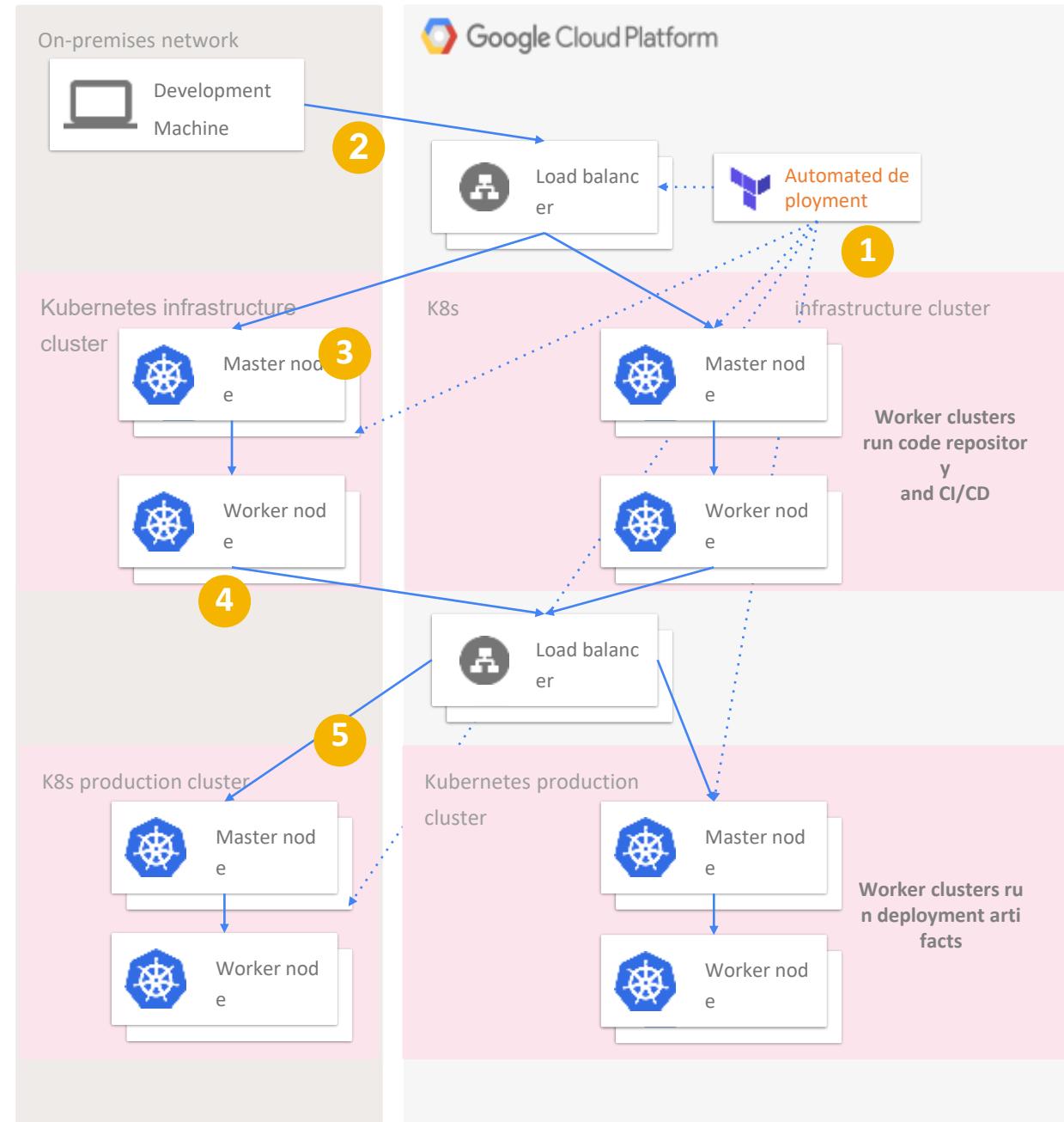
- 1 Terraform provisions necessary resources
- 2 Developer pushes code to repo
- 3 Master node delegates to worker node
- 4 Code is tested and built by worker node
- 5 Application is deployed to production environment

## Advantages

- Redundant
- Hosted code repo, load balancer, and multiple master clusters
- No single point of failure

## Disadvantages

- You have to manage everything yourself



# Cloud-managed CI/CD with Kubernetes

GCP NoOps Managed Services solution. Code repository, Load Balancer, and Deployment Manager all run as managed services.

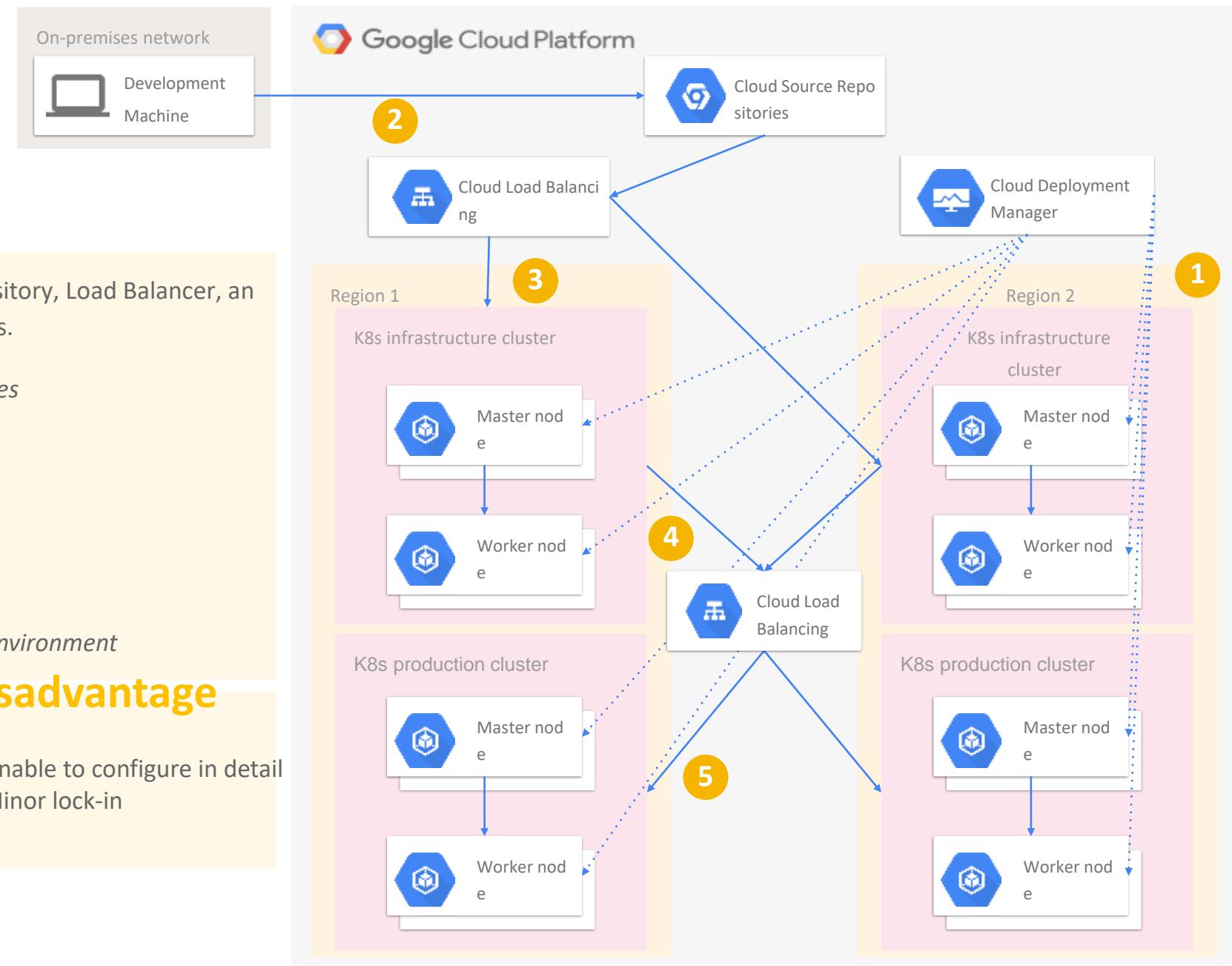
- 1 Terraform provisions necessary resources
- 2 Developer pushes code to repo
- 3 Master node delegates to worker node
- 4 Code is tested and built by worker node
- 5 Application is deployed to production environment

## Advantages

- High availability
- Good use of NoOps

## Disadvantage

- Unable to configure in detail
- Minor lock-in



## Module 8

# Multi-Cloud에서의 IAM / Security / Compliance 관리

**Part I**

**IAM**

# 관리/보안 서비스



## 관리 및 거버넌스

AWS Organizations  
CloudWatch  
AWS Auto Scaling  
CloudFormation  
CloudTrail  
Config  
OpsWorks  
Service Catalog  
Systems Manager  
AWS AppConfig  
Trusted Advisor  
Control Tower  
AWS License Manager  
AWS Well-Architected Tool  
Personal Health Dashboard ↗  
AWS Chatbot  
Launch Wizard

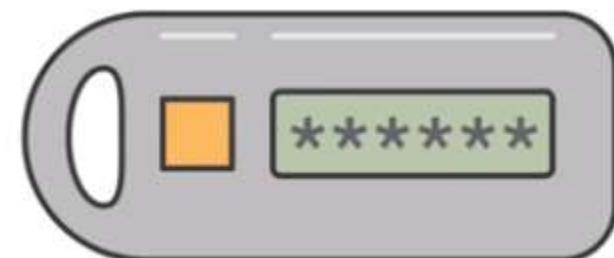


## 보안, 자격 증명 및 규정 준수

IAM  
Resource Access Manager  
Cognito  
Secrets Manager  
GuardDuty  
Inspector  
Amazon Macie ↗  
AWS Single Sign-On  
Certificate Manager  
Key Management Service  
CloudHSM  
Directory Service  
WAF & Shield  
Artifact  
Security Hub  
Detective

# **AWS IAM MFA**

### MFA(Multi-Factor Authentication)



- 물리적 기반 또는 소프트웨어 기반 토큰 통한 멀티팩터 인증
- IAM 사용자 및 루트 계정에서 활성화

# AWS IAM : 다중 인증

The screenshot shows the AWS IAM console with the following details:

- Identity and Access Management(IAM)** is selected in the sidebar.
- 대시보드** (Dashboard) is the active section.
- IAM 사용자 로그인 링크:** <https://697559359741.signin.aws.amazon.com/console>
- IAM 리소스** statistics:
  - 사용자: 1
  - 역할: 6
  - 그룹: 1
  - 고객 관리형 정책: 0
  - 자격 증명 공급자: 0
- 보안 상태**: 3/5 완료
  - 루트 액세스 키 삭제** (checked)
  - 루트 계정에서 MFA 활성화** (warning icon)
  - AWS 루트 계정에서 멀티 팩터 인증(MFA)을 활성화하여 다른 보호 단계를 추가하면 계정을 안전하게 유지하는 데 도움이 됩니다. [자세히 알아보기](#)
  - MFA 관리** button
  - 개별 IAM 사용자 생성** (checked)
  - 그룹을 사용하여 권한 할당** (checked)
  - IAM 비밀번호 정책 적용** (warning icon)

참고: [https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_credentials\\_mfa.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa.html)

# AWS IAM : 다중 인증

The screenshot shows the AWS IAM Multi-Factor Authentication (MFA) page. The left sidebar lists various IAM management options. The main content area is titled "보안 자격 증명" (Security Credentials) and provides information about MFA for AWS environments. It includes sections for "비밀번호" (Passwords), "멀티 팩터 인증(MFA)" (Multi-Factor Authentication), "액세스 키(액세스 키 ID 및 비밀 액세스 키)" (Access Keys), "CloudFront 키 페어" (CloudFront Key Pair), "X.509 인증서" (X.509 Certificate), and "계정 ID" (Account ID). A blue button labeled "MFA 활성화" (Enable MFA) is visible under the MFA section.

aws 서비스 리소스 그룹 ★

Identity and Access Management(IAM)

보안 자격 증명

이 페이지를 사용하여 AWS 계정의 자격 증명을 관리합니다. AWS IAM(Identity and Access Management) 사용자에 대한 자격 증명을 관리하려면 [IAM 콘솔](#) 을(들) 사용하십시오.

AWS 자격 증명 유형과 사용 방법에 대해 자세히 알아보려면 AWS 일반 참조의 [AWS 보안 자격 증명](#) 을(들) 참조하십시오.

▲ 비밀번호

▼ 멀티 팩터 인증(MFA)

MFA를 사용하여 AWS 환경의 보안을 강화합니다. MFA 보호 계정에 로그인하려면 사용자 이름, 암호, MFA 디바이스에서 받은 인증 코드가 필요합니다.

[MFA 활성화](#)

▲ 액세스 키(액세스 키 ID 및 비밀 액세스 키)

▲ CloudFront 키 페어

▲ X.509 인증서

▲ 계정 ID

대시보드

액세스 관리

그룹

사용자

역할

정책

자격 증명 공급자

계정 설정

보고서 액세스

액세스 분석기

아카이브 규칙

분석기 세부 정보

devops\_test 글로벌 지원

# AWS IAM : 다중 인증

## MFA 디바이스 관리

할당할 MFA 디바이스의 유형 선택:

- 가상 MFA 디바이스**  
모바일 디바이스 또는 컴퓨터에 설치된 Authenticator 앱
- U2F 보안 키**  
YubiKey 또는 기타 호환 U2F 디바이스
- 다른 하드웨어 MFA 디바이스**  
Gemalto 토큰

지원되는 MFA 디바이스에 대한 자세한 내용은 [AWS MFA\(Multi-Factor Authentication\)](#) 을(를) 참조하십시오.

[취소](#) [계속](#)

# AWS IAM : 다중 인증

가상 MFA 디바이스 설정

1. 모바일 디바이스 또는 컴퓨터에 호환 앱 설치  
[호환 애플리케이션 목록 참조](#)

2. 가상 MFA 앱 및 디바이스의 카메라를 사용하여 QR 코드 스캔



또는 비밀 키를 입력할 수 있습니다. [비밀 키 표시](#)

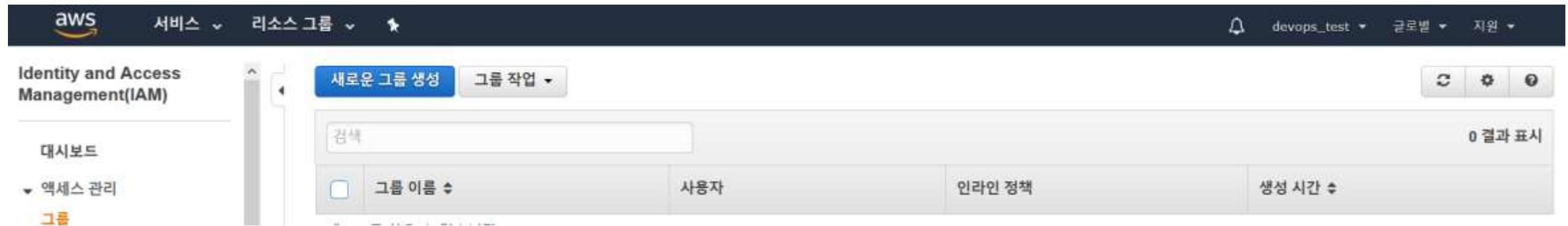
3. 아래에 2개의 연속된 MFA 코드 입력

MFA 코드 1

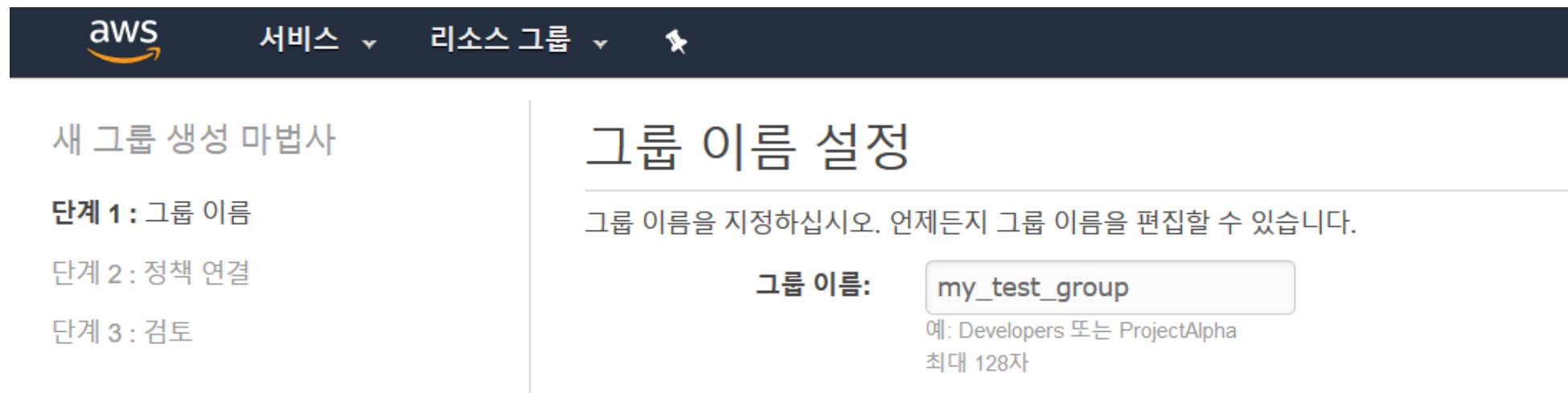
MFA 코드 2

[취소](#) [이전](#) [MFA 할당](#)

# AWS IAM : 그룹 생성 및 정책 적용



The screenshot shows the AWS IAM Groups management interface. At the top, there's a navigation bar with the AWS logo, '서비스', '리소스 그룹', and other account-related options. Below the navigation is the 'Identity and Access Management(IAM)' section header. On the left, a sidebar lists '대시보드', '액세스 관리', and '그룹'. The main area has tabs for '새로운 그룹 생성' (highlighted in blue) and '그룹 작업'. A search bar labeled '검색' is at the top. Below it is a table with columns: '그룹 이름' (with a dropdown arrow), '사용자', '인라인 정책', and '생성 시간'. A '0 결과 표시' message is on the right. The overall theme is dark with light-colored UI elements.



This screenshot shows the first step of the 'New Group Wizard' titled '새 그룹 생성 마법사'. The left sidebar lists three steps: '단계 1: 그룹 이름' (selected), '단계 2: 정책 연결', and '단계 3: 검토'. The main panel is titled '그룹 이름 설정' and contains the instruction '그룹 이름을 지정하십시오. 언제든지 그룹 이름을 편집할 수 있습니다.' Below this is a text input field labeled '그룹 이름:' containing 'my\_test\_group'. A note below the input says '예: Developers 또는 ProjectAlpha' and '최대 128자'. The overall design is clean with a white background and blue links.

# AWS IAM : 그룹 생성 및 정책 적용

aws 서비스 리소스 그룹 ★

devops\_test 글로벌 지원

새 그룹 생성 마법사

단계 1: 그룹 이름

단계 2: 정책 연결

단계 3: 검토

정책 연결

연결할 정책을 하나 이상 선택하십시오. 각 그룹에는 최대 10개의 정책이 연결될 수 있습니다.

필터: 정책 유형 ▾ power 7 결과 표시

	정책 이름	연결된 개체	생성 시간
<input checked="" type="checkbox"/>	PowerUserAccess	1	2015-02-07 03:39 UTC+0900
<input type="checkbox"/>	AmazonCognitoPowerUser	0	2015-03-25 02:14 UTC+0900
<input type="checkbox"/>	AmazonEC2ContainerRegistryPowerUser	0	2015-12-22 02:05 UTC+0900
<input type="checkbox"/>	AWSCodeCommitPowerUser	0	2015-07-10 02:06 UTC+0900
<input type="checkbox"/>	AWSDataPipeline_PowerUser	0	2017-01-20 08:16 UTC+0900
<input type="checkbox"/>	AWSKeyManagementServicePowerUser	0	2015-02-07 03:40 UTC+0900

aws 서비스 리소스 그룹 ★

새 그룹 생성 마법사

단계 1: 그룹 이름

단계 2: 정책 연결

단계 3: 검토

검토

다음 정보를 검토한 다음, 그룹 생성을 클릭하여 계속하십시오.

그룹 이름	my_test_group	그룹 이름 편집
정책	arn:aws:iam::aws:policy/PowerUserAccess	정책 편집

# AWS Organization

# AWS Organization

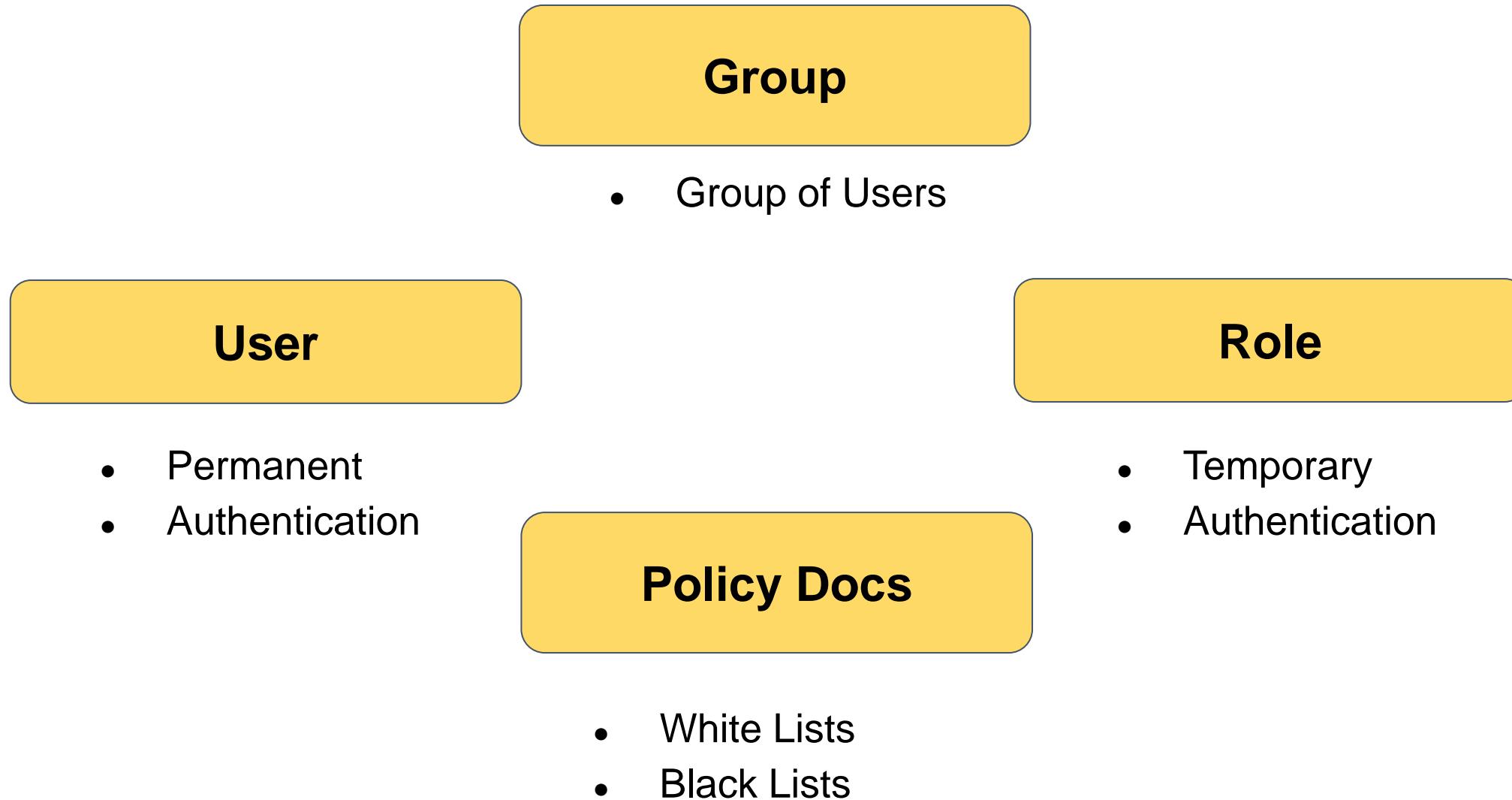
## Introduction

- 모든 AWS 계정의 중앙 집중식 관리
- 모든 멤버 계정에 대한 통합 결제
  - 조직의 마스터 계정을 사용하여 모든 멤버 계정을 통합하고 요금을 지불 가능
- 예산, 보안, 규정 준수 필요 충족을 위한 계정의 계층적 그룹화
  - 계정을 조직 단위(OU)로 그룹화하고 OU마다 다른 액세스 정책을 연결 가능
- 각 계정이 액세스할 수 있는 AWS 서비스 및 API 작업 제어
- AWS Identity and Access Management(IAM)에 대한 통합 및 지원
- 다른 AWS 서비스와의 통합
- 최종 일관 데이터 복제
- AWS Organizations는 추가 비용 없이 제공
- AWS Organizations에 액세스 : Management Console, 명령줄 도구(CLI), SDK, HTTPS 쿼리 API

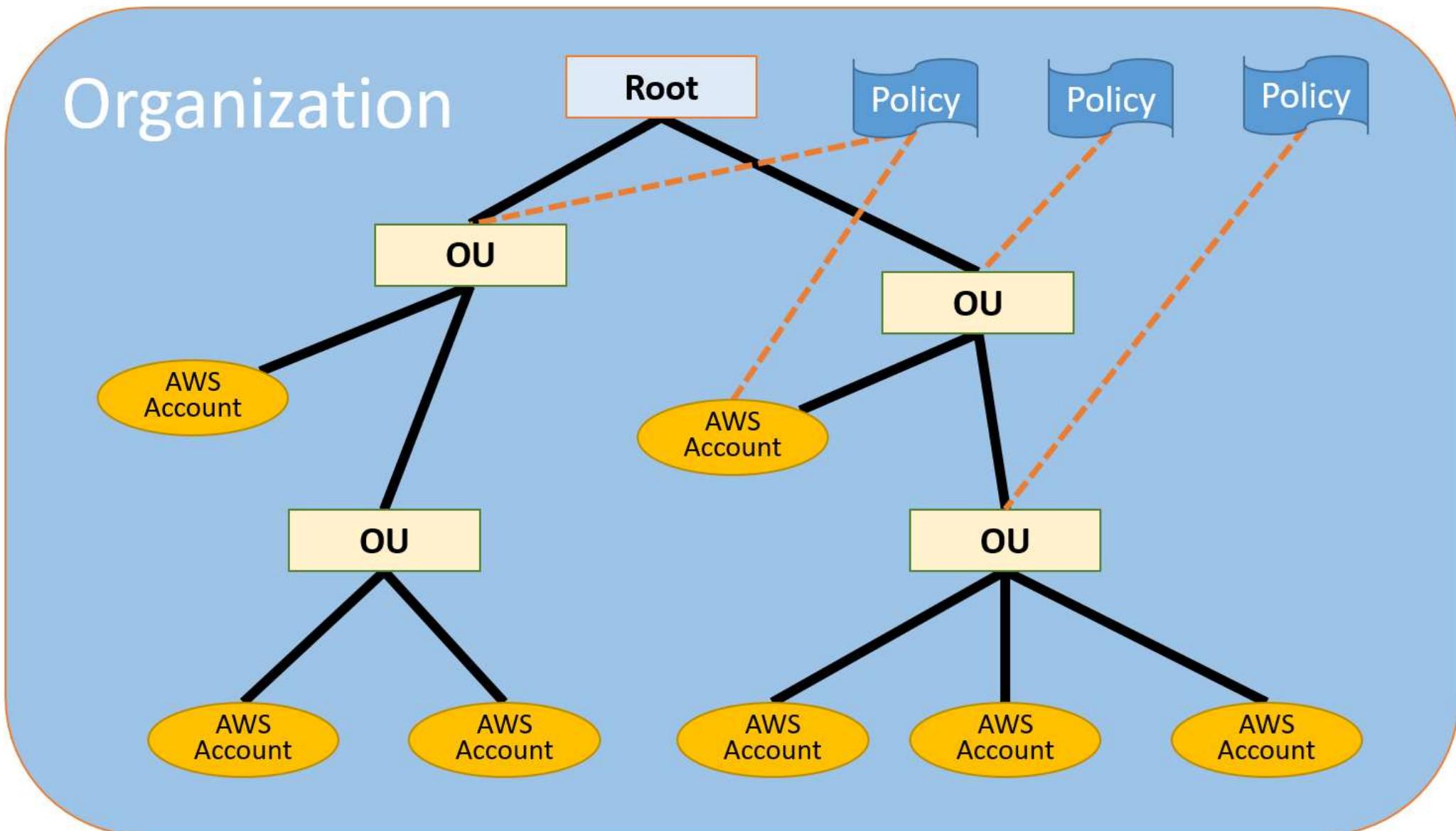
[https://docs.aws.amazon.com/ko\\_kr/organizations/latest/userguide/orgs\\_introduction.html](https://docs.aws.amazon.com/ko_kr/organizations/latest/userguide/orgs_introduction.html)

# AWS Organization

## 기본 개념



# AWS Organization



# AWS Organization

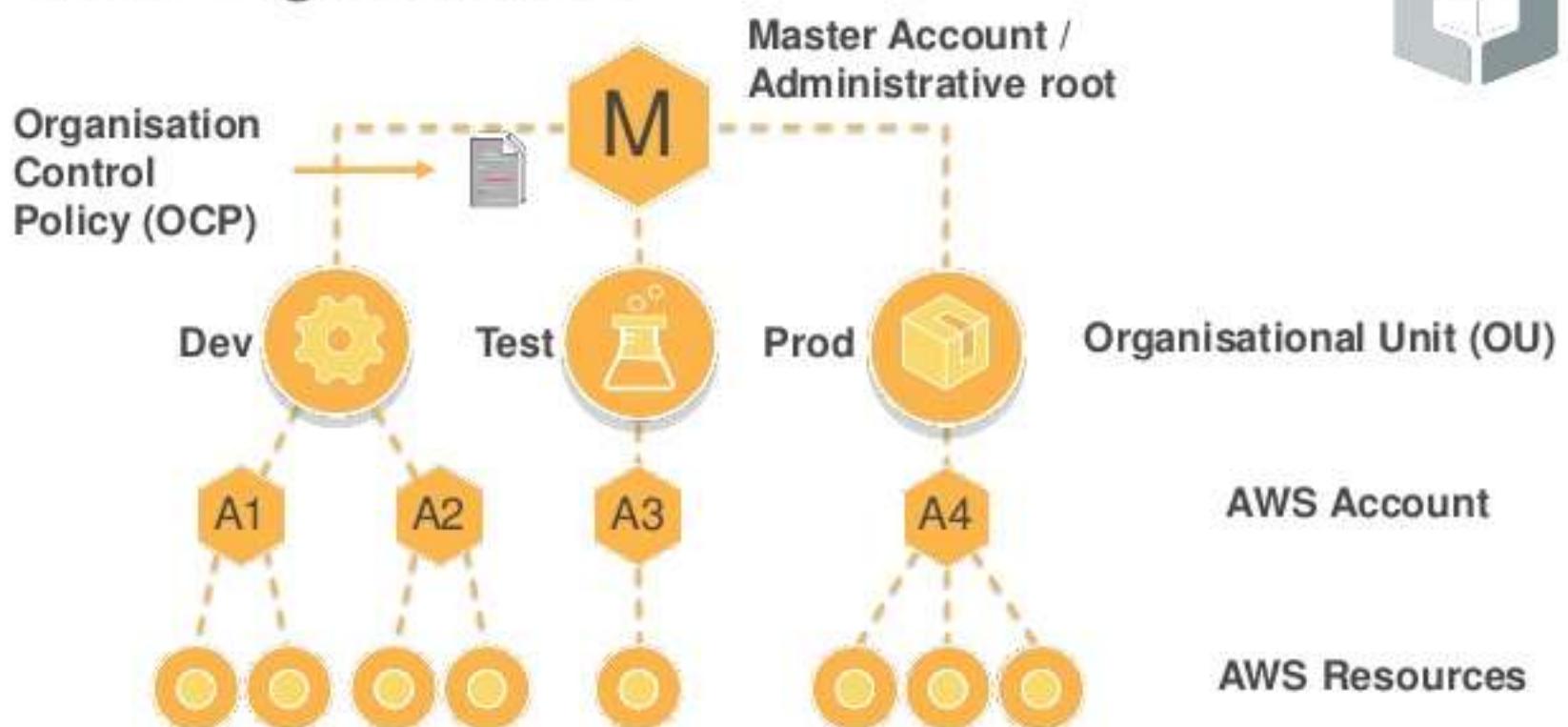
## AWS Cost and Usage Report

- AWS Cost and Usage report는 AWS 사용 내역을 추적하고 AWS 계정의 예상 요금을 알림
- AWS Organizations의 통합 결제 기능을 사용하는 경우, 이 보고서는 마스터 계정에서만 사용 할 수 있으며 마스터 계정에 연결된 모든 멤버 계정의 활동이 포함
- AWS는 계정에서 본인이 지정한 Amazon S3 버킷으로 보고서 파일을 전송하고 하루 최대 3 회 보고서를 업데이트
- AWS는 매달 말에 보고서를 완료합니다. 완성된 보고서에는 혼합 요금과 일반 요금의 계산 결과뿐만 아니라 그 달의 모든 사용 내역이 포함
- AWS가 청구서 금액이 확정된 후 해당 월에 대해 환급, 크레딧 및 지원 이용 요금을 적용하는 경우에는 완성된 이후에 AWS 보고서가 업데이트
- AWS는 계정당 5 AWS 비용 및 사용 보고서를 지원합니다. 이 보고서는 무료지만, 스탠다드 Amazon S3 스토리지 요금이 적용

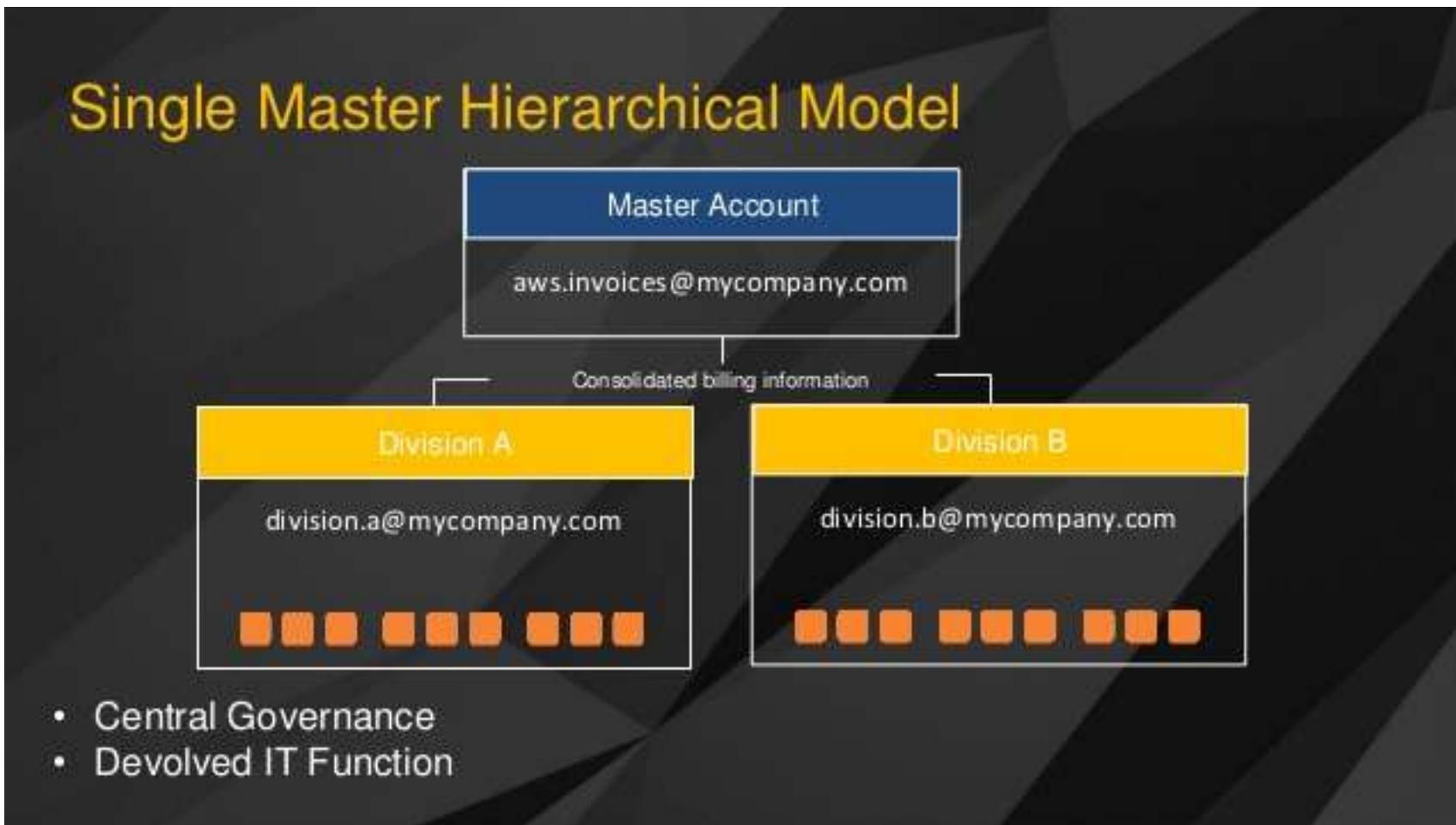
[https://docs.aws.amazon.com/ko\\_kr/awsaccountbilling/latest/aboutv2/billing-reports-costusage.html](https://docs.aws.amazon.com/ko_kr/awsaccountbilling/latest/aboutv2/billing-reports-costusage.html)

# AWS Organization

## AWS Organisations



# AWS Organization



# **Part II**

# **Security & Compliance**

# **AWS Security & Compliance**

# AWS 보안

- AWS 클라우드 보안 : 전략적 보안
- AWS는 보안, 고성능, 복원력 및 효율성을 갖춘 애플리케이션용 인프라를 구축하는 데 도움이 되도록 설계



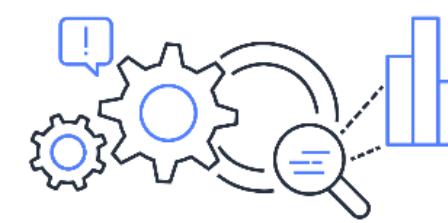
방지

원활하고 계획에 따른 AWS 채택 전략을 위해 사용자 권한 및 자격 증명, 인프라 보호 및 데이터 보호 수단을 정의합니다.



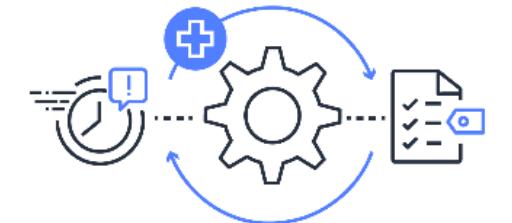
탐지

로깅 및 모니터링 서비스를 통해 조직의 보안 상태를 파악할 수 있습니다. 이 정보는 이벤트 관리, 테스트 및 감사를 위해 확장 가능한 플랫폼에 수집됩니다.



대처

자동화된 인시던트 응답 및 복구를 통해 보안 팀이 단순히 문제에 대처하는 데 그치지 않고 근본 원인을 분석할 수 있도록 지원합니다.



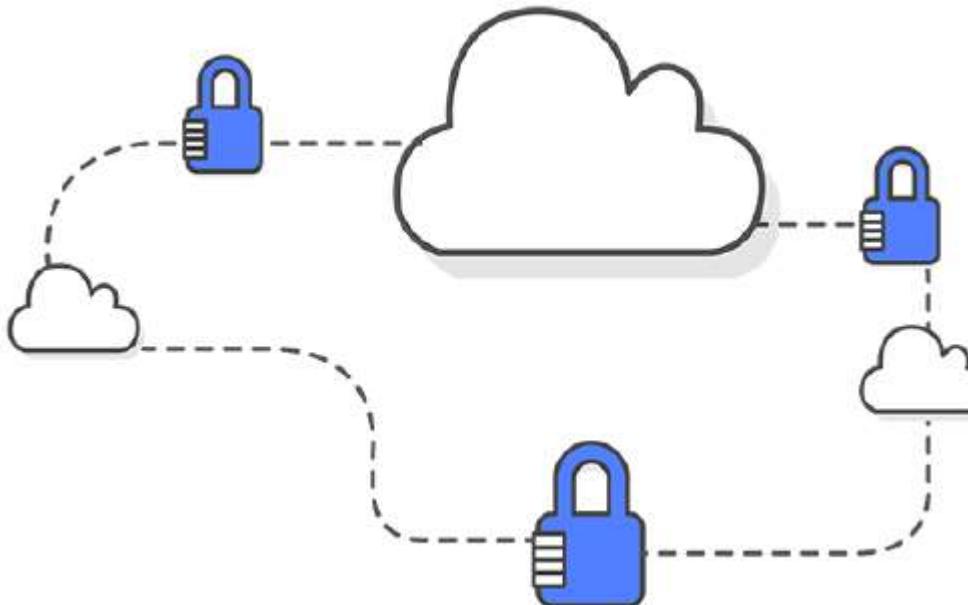
해결

이벤트 기반 자동화를 활용하여 거의 실시간으로 신속하게 문제를 해결하고 AWS 환경을 보호합니다.

# AWS 보안

## 보안 구현 요소

- 보안에 대한 접근 방식 (Access, Role, Policy)
- AWS 환경 제어
- AWS 제품 및 기능



# AWS 보안

- AWS 보안 / 컴플라이언스 소개

30개 이상의 글로벌  
인증 및 자격 취득/준수



보안에 민감한 군사 조직과 정보기관  
글로벌 은행, 기업에서 이미 구축하였고  
보안에 만족하고 있음

추가 비용없이 강력한  
기본 기능 및 도구 제공



AWS는 업계를 리딩하는 혜택 제공  
Benefit from AWS industry leading  
보안팀은 24/7, 365일 운영/지원

"AWS와 긴밀한 협조하에 보안 모델을 개  
발하여, 우리 자체 데이터센터에서 하는 것  
보다 더 안전하게 퍼블릭 클라우드를 운영  
할 수 있게 되었습니다"

1백만 이상 고객 경험에서  
수집/활용 보안 강화

Rob Alexander - CIO, Capital One

# AWS 보안

- AWS 보안 컴플라이언스 인증



## 디자인 원리 (1)

- Implement a strong identity foundation
  - Implement the principle of least privilege and enforce separation of duties with appropriate authorization for each interaction with your AWS resources.
- Enable traceability
  - Monitor, alert, and audit actions and changes to your environment in real time. Integrate logs and metrics with systems to automatically respond and take action.
- Apply security at all layers
  - Rather than just focusing on protection of a single outer layer, apply a defense-in-depth approach with other security controls.
- Automate security best practices
  - Automated software-based security mechanisms improve your ability to securely scale more rapidly and cost effectively. Implement controls that are defined and managed as code in version-controlled templates.

## 디자인 원리 (2)

- Protect data in transit and at rest
  - Classify your data into sensitivity levels and where appropriate, use mechanisms like encryption and access control.
- Enforce the principle of least privilege
  - Access to data should only be granted to the people who really need that access. Start with denying access to everything and grant access as needed.
- Prepare for security events
  - Prepare for an incident by having an incident management process that aligns to your organizational requirements. Run incident response simulations and use tools with automation to increase your speed for detection, investigation, and recovery.

## 네트워크 보안

- 내장 방화벽
- 전송 중 암호화
- 프라이빗 전용 연결
- DDoS 완화



## 인벤토리 및 구성 관리

- 배포 도구
- 인벤토리 및 구성 도구
- 템플릿 정의 및 관리 도구

## 데이터 암호화

- 암호화 기능
- 키 관리 옵션
  - AWS Key Management Service
- 하드웨어 기반 암호화 키 스토리지 옵션
  - AWS CloudHSM



## 액세스 제어 (Access Control) 및 관리

- Identity and Access Management (IAM)
- Multi factor authentication (MFA)
- 기업 디렉터리와 통합 및 연동
- Amazon Cognito
- AWS SSO



# AWS 보안

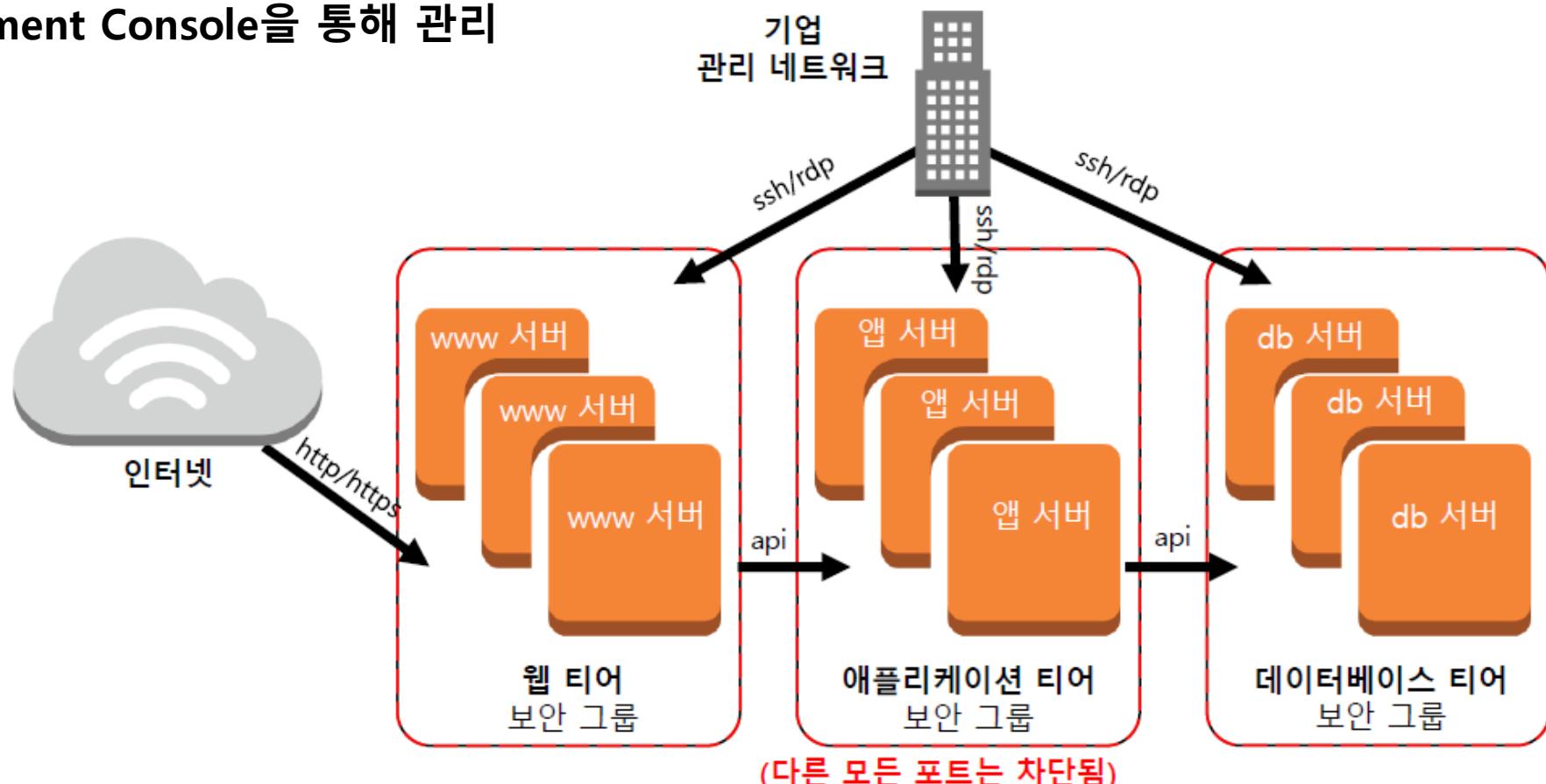
## 모니터링 및 로깅

- 위협요소를 낮출 수 있는 도구 및 기능
- API 호출에 대한 심층적인 가시성
- 로그 집계 및 옵션
- 경고 알림
  - CloudWatch
  - CloudTrail

# AWS 보안

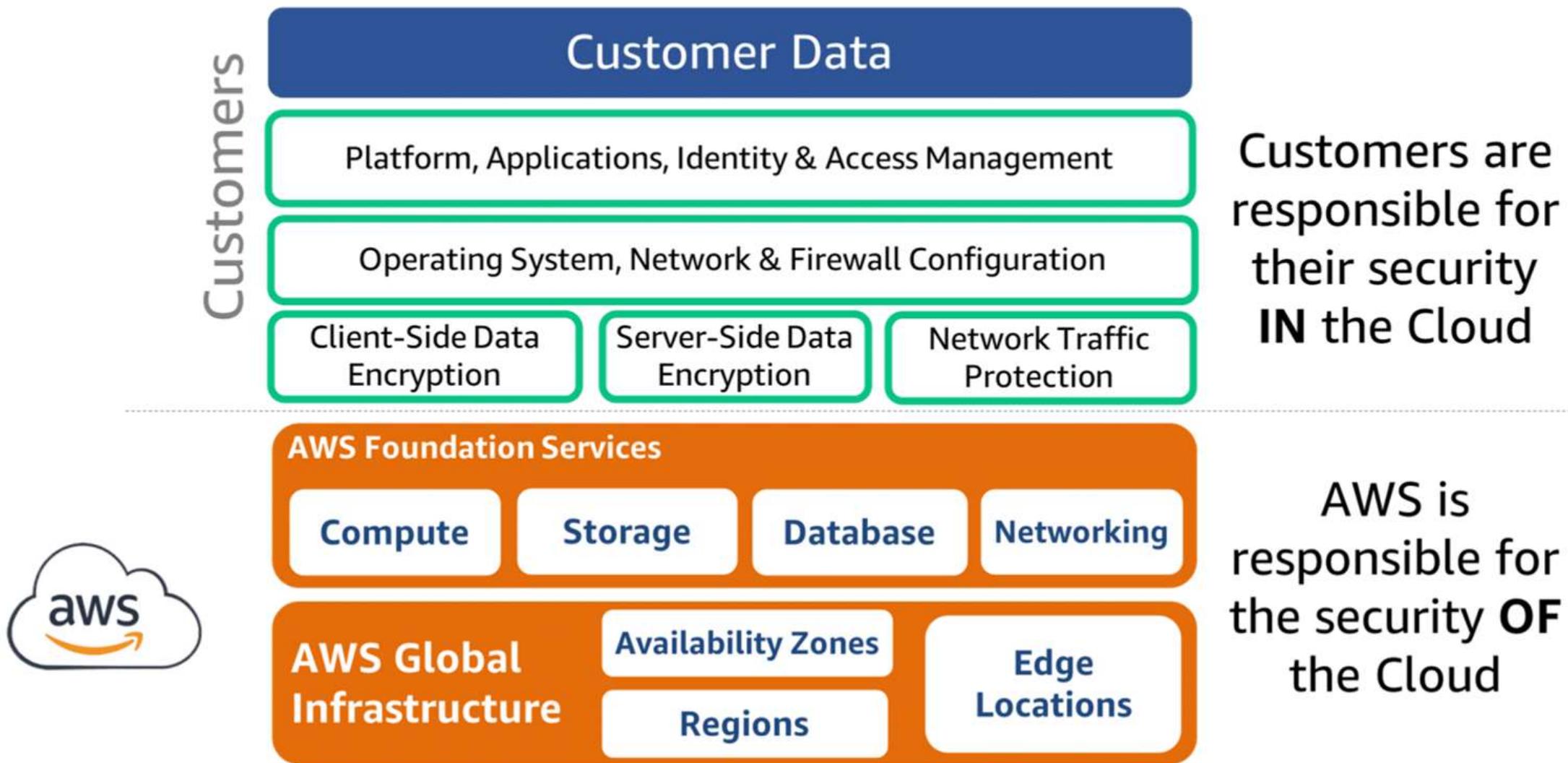
## 보안 그룹 (EC2)

- 내장된 가상의 방화벽 역할
- 규칙을 통해 EC2 Instance에 대한 Access 제어
- AWS Management Console을 통해 관리



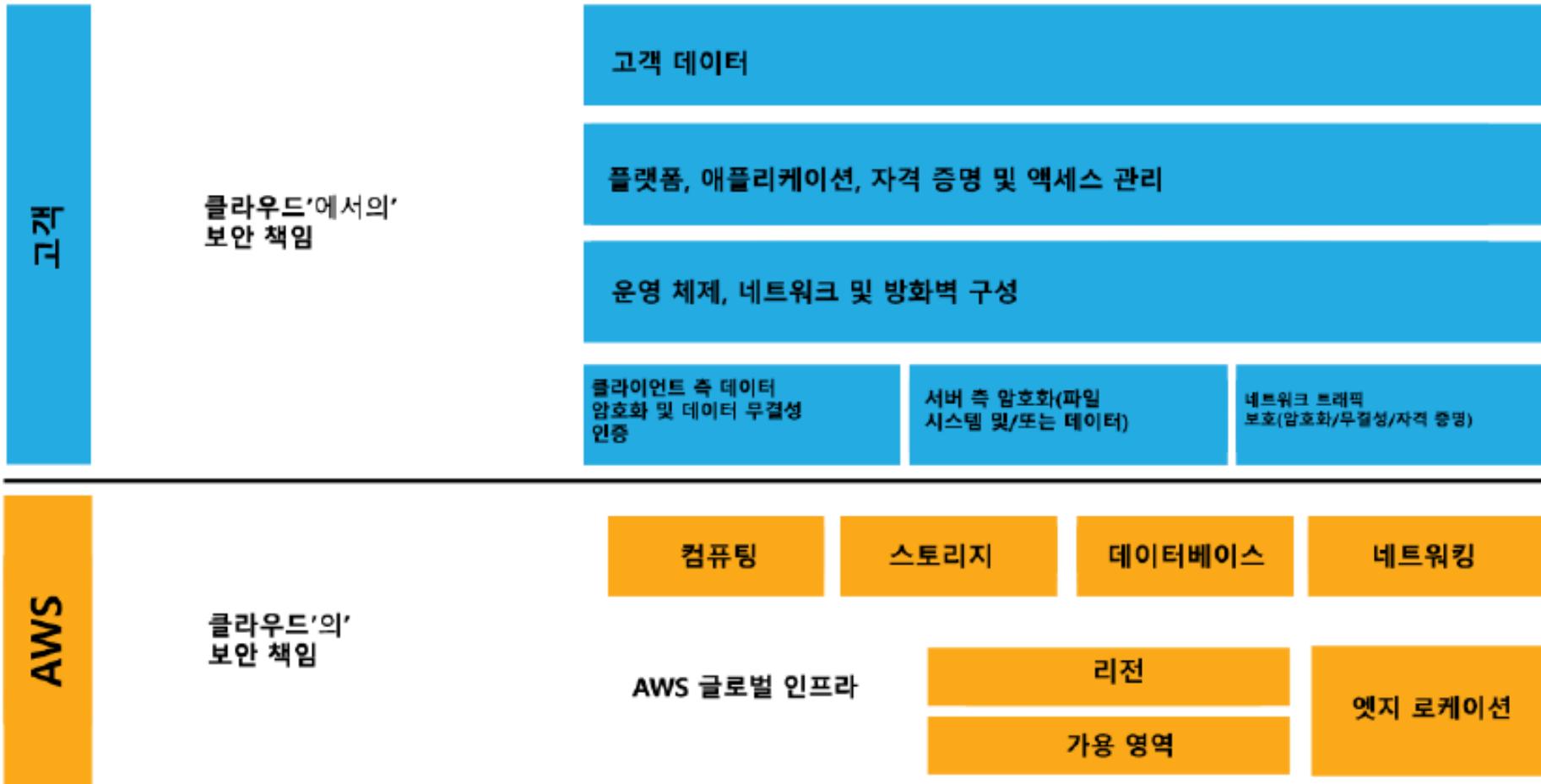
# AWS 보안

## 공동책임 모델 (Shared Responsibility Model)



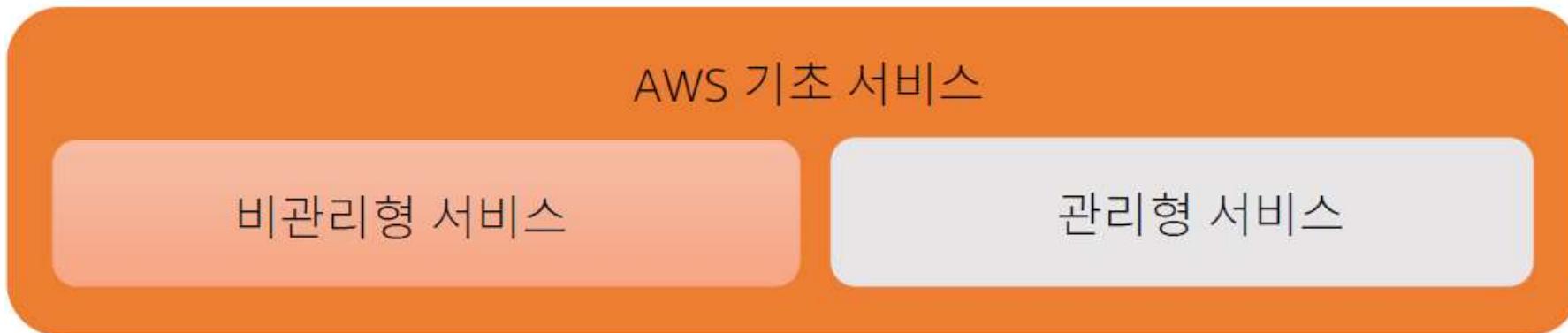
# AWS 보안

## 공동책임 모델 (Shared Responsibility Model)



# AWS 보안

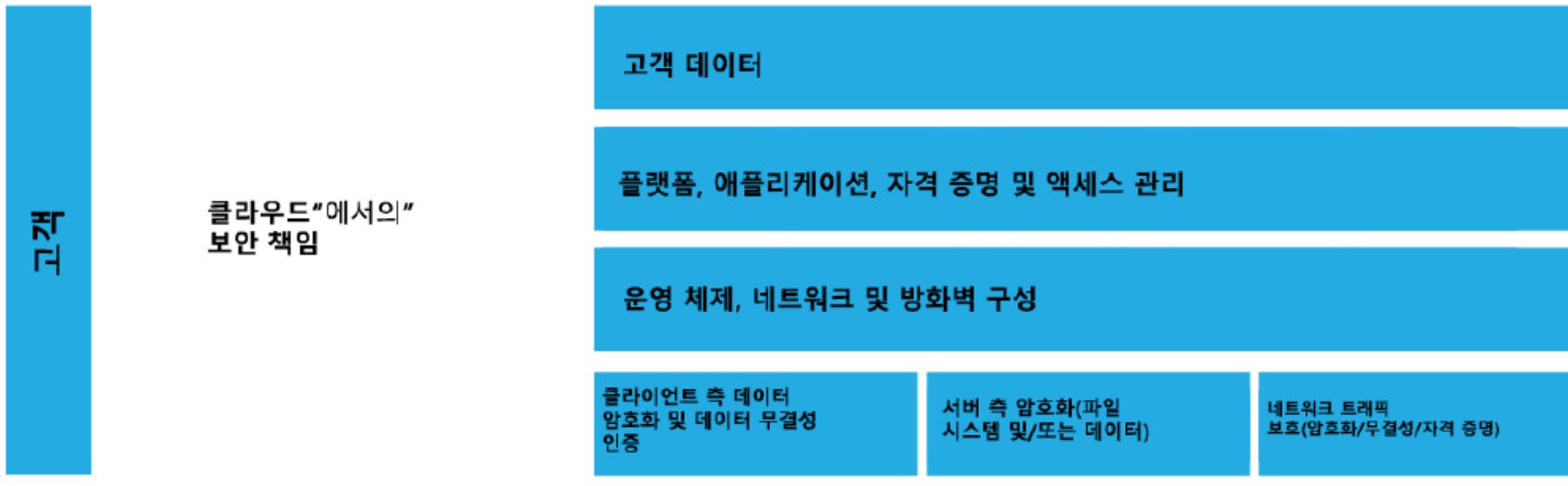
## 공동책임 모델 (Shared Responsibility Model)



- Amazon EC2
- Amazon EBS
- 상속된 제어 항목
  - 물리적
  - 환경
- 공동 제어
  - 패치 관리
  - 구성 관리
  - 인식 및 교육
- 고객 특정
  - 서비스/통신 보호
  - 영역 보안

# AWS 보안

## 공동책임 모델 (Shared Responsibility Model)



- 무엇을 저장할지
- 어떤 AWS 서비스를
- 어느 위치에서
- 고객이 제어권 유지
- 서비스에 따라 모델이 달라짐
- 어떤 콘텐츠 형식 및 구조로
- 누구에게 액세스 권한이 있는지

# AWS 보안

## 공동책임 모델 (Shared Responsibility Model)



© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.



# 참고 : AWS 보안

## 클라우드 보안 사고

- 클라우드 보안 관련 사고 중 **95%**가 사용자 부주의와 관련 (Gartner)
- 사용자의 계정 정보를 탈취하거나, 사용 후 부주의로 열어놓은 접속권한을 이용
- 개발 후 정리가 덜 된 자원이나 퇴사자, 협력업체 사용 자원 등의 관리가 어려움  
→ 가상 자원의 남용 (암호화폐 채굴 등), 데이터 탈취 등의 사고로 이어짐
- S사 사고 사례

Forbes

### Samsung Investigates Massive Data Leak -- What You Need To Know

May 9, 2019



<https://www.forbes.com/sites/daveywinder/2019/05/09/samsung-investigates-massive-data-leak-what-you-need-to-know/#2321fbf32e2c>

# 참고 : AWS 보안

## Data Center Security



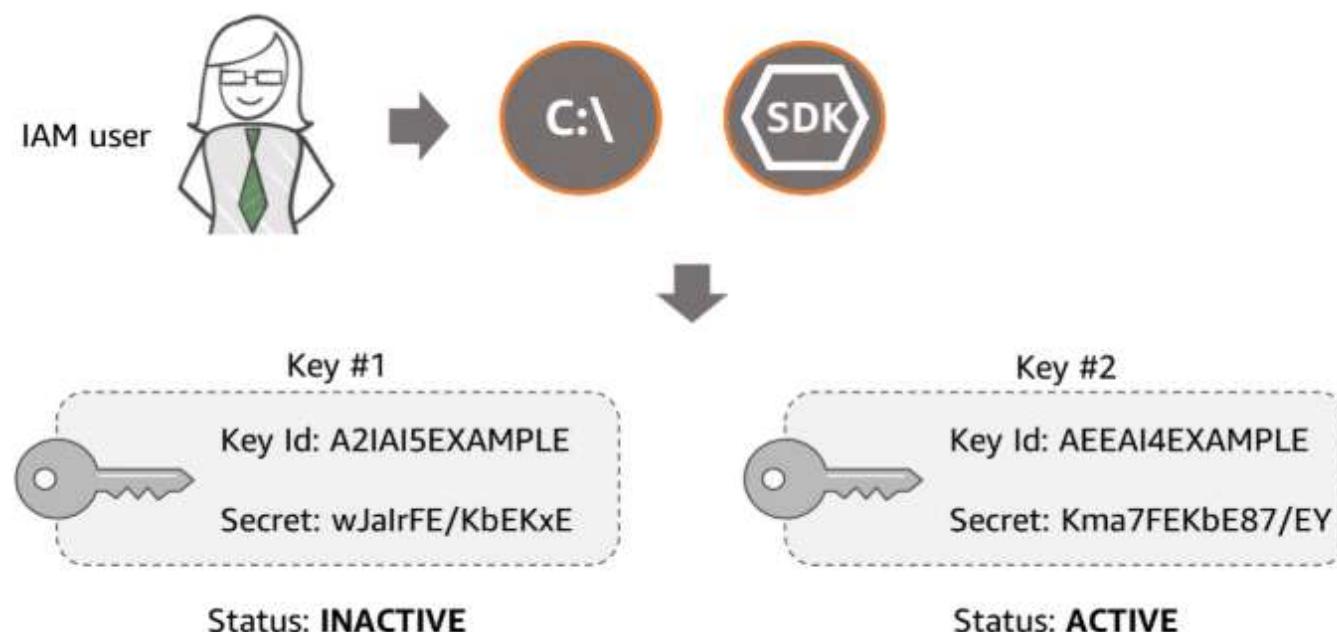
- Perimeter Layer
- Environmental Layer
- Infrastructure Layer
- Data Layer
- Resources

<https://aws.amazon.com/ko/compliance/data-center/data-centers/>

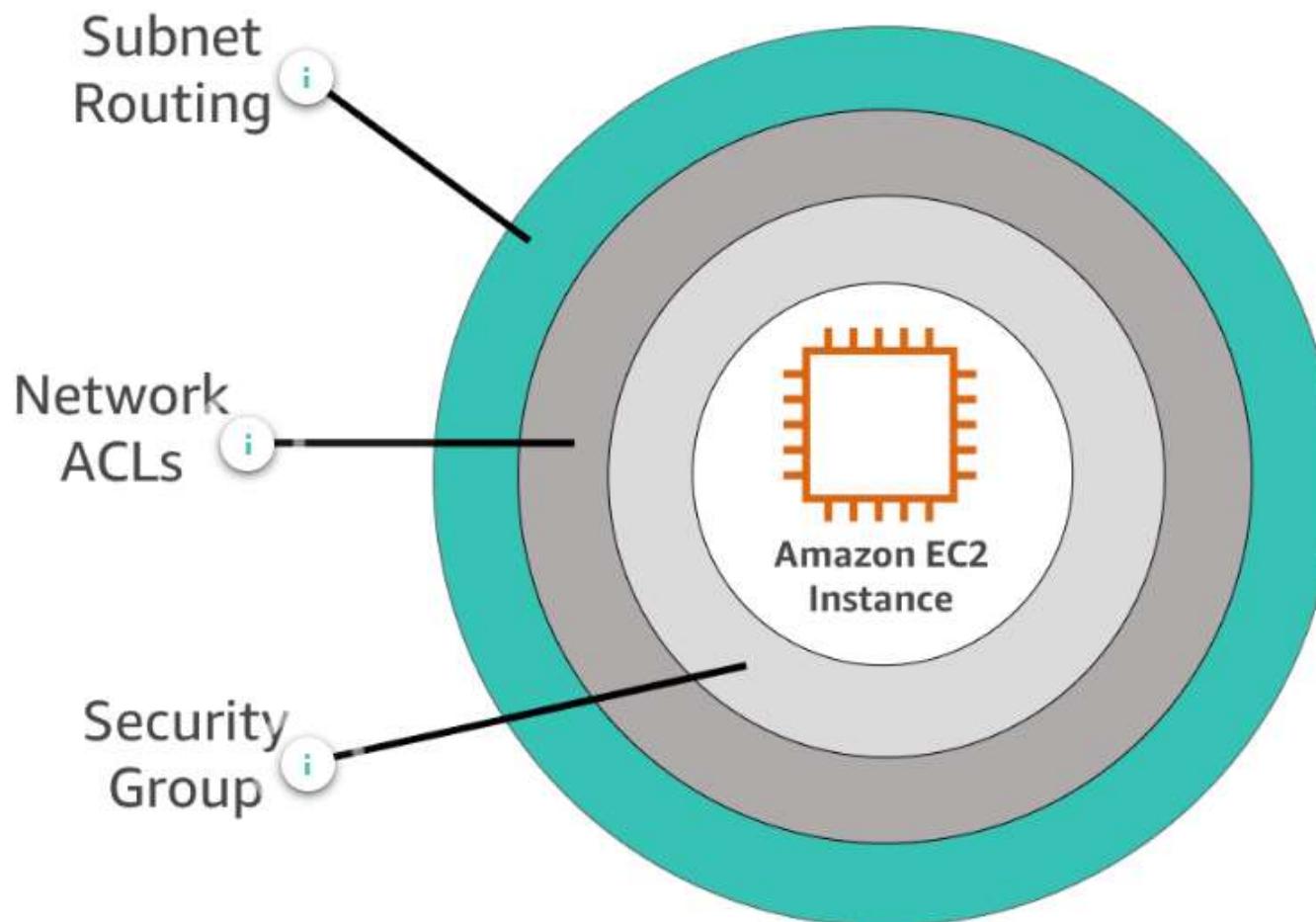
# 참고 : AWS 보안

## AWS Credentials : User Access Keys

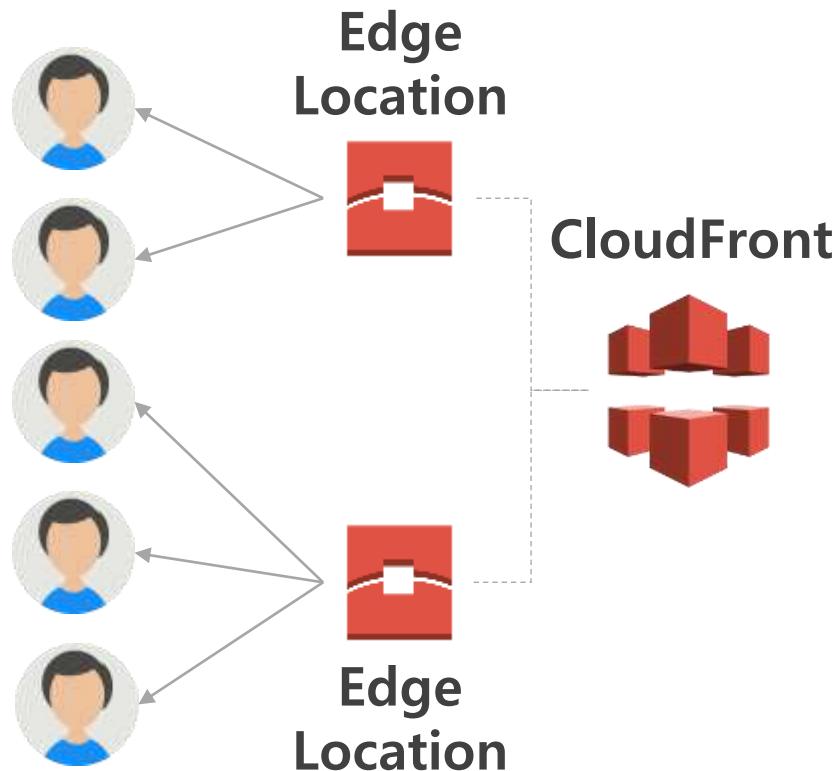
Users need their own access keys to make programmatic calls to AWS using the AWS CLI, the AWS SDKs, or direct HTTPS calls using the APIs for individual AWS services. Access keys are used to digitally sign API calls made to AWS services. Each access key credential is comprised of an access key ID and a secret key. Each user can have two active access keys, which is useful when you need to rotate the user's access keys or revoke permissions.



## Infrastructure Protection via Isolation



# 참고 : CloudFront



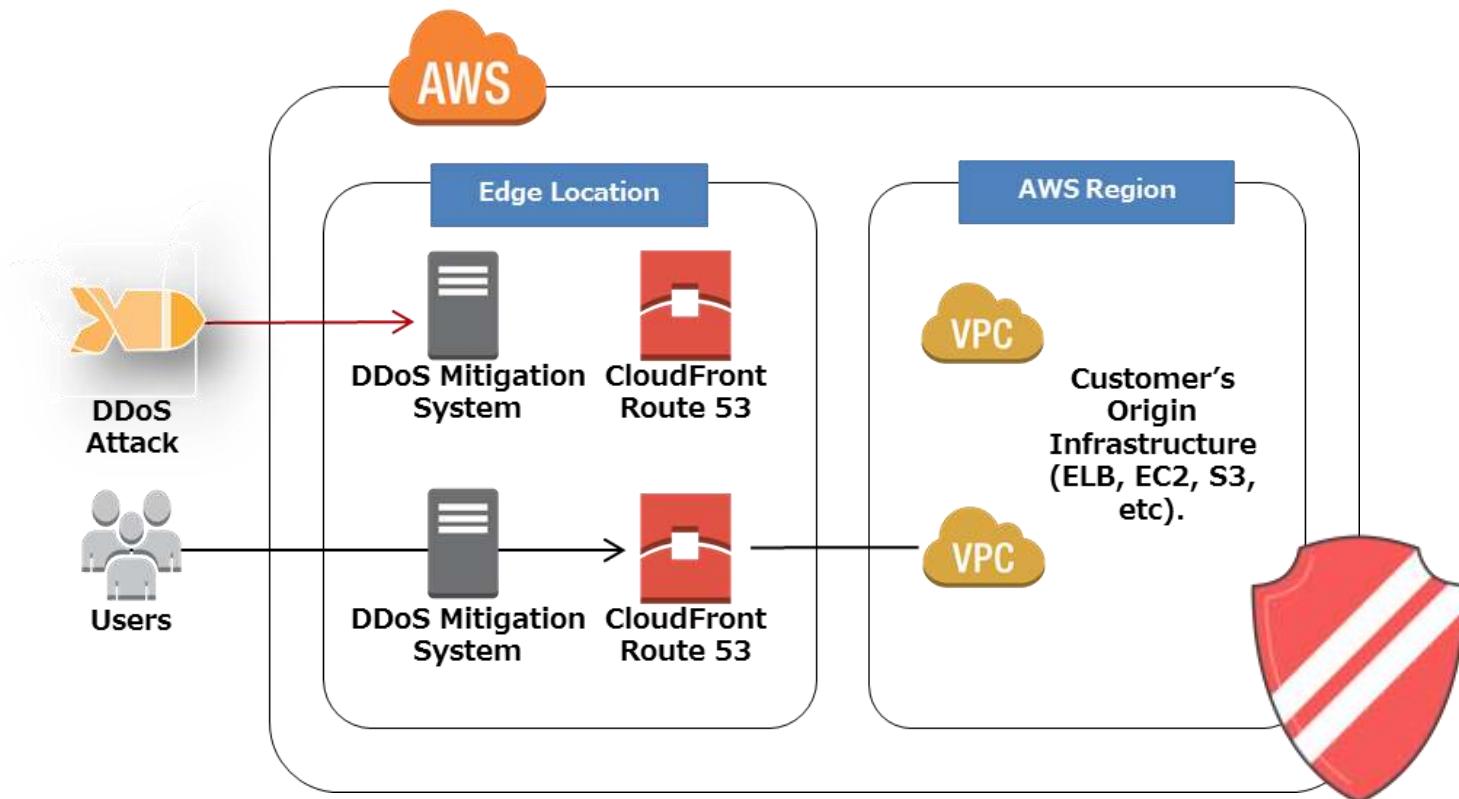
- 컨텐츠 (이미지, HTML 등)를 캐싱하여 성능 가속
- 전 세계 113개 엣지 로케이션: 글로벌 서비스
- AWS 서비스 → CloudFront 간 데이터 전송 무료
- 글로벌 고속 백본 네트워크 확보
- DDoS 방어 무료 제공 (AWS Shield Standard)

# 참고 : Amazon CloudFront: Layer 3/4 DDoS 방어 제공

“AWS Shield Standard에 의한 L3 / L4 DDoS 보호는 추가 비용 없이 포함”



AWS  
Shield  
Standard



# **GCP Security & Compliance**

# 구글클라우드 보안 협력

## Security Collaboration

- Google은 infrastructure 보안에 책임
- 사용자는 데이터 보안에 책임
- Google은 사용자에게 best practices, templates, 제품, 솔루션 등을 제공하여 보안을 도와줌.

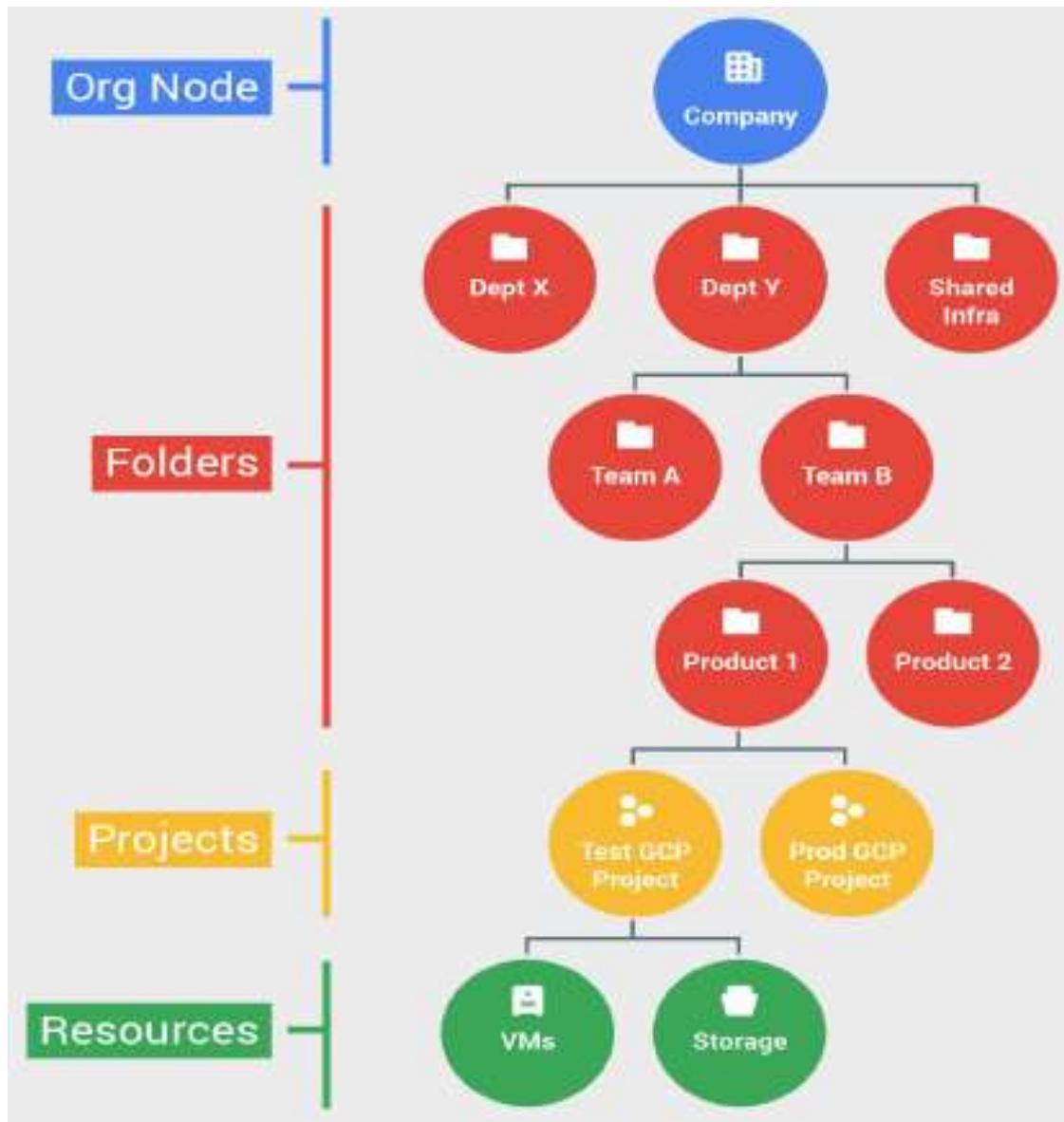
█ Customer-managed   █ Google-managed

Responsibility	On-premises	Infrastructure as a Service	Platform as a Service	Managed services
Content				
Access policies				
Usage				
Deployment				
Web application security				
Identity				
Operations				
Access and authentication				
Network security				
OS, data, and content				
Audit logging				
Network				
Storage and encryption				
Hardware				

# 구글클라우드 자원 계층체계 (Resource Hierarchy)

자원의 계층구조 수준(Resource hierarchy levels)이 신뢰 경계선(trust boundary)을 결정

- 조직도에 따라 자원을 그룹화
- 계층 구조의 수준은 자원의 고립(isolation)과 신뢰 경계선(trust boundary)을 제공
- 상위 수준의 정책을 자동적으로 승계



# 구글클라우드 프로젝트 (Project)

모든 GCP 서비스는 프로젝트와 연계되어서 비용을 처리

- 자원 추적 및 가능 사용량(Quota) 할당
- 사용료 정산 (Billing)
- 권한(Permission) 및 인증서(Credential) 관리
- 서비스 사용 권한 및 API 사용 권한 설정

프로젝트는 아래와 같은 3가지 자격증명 속성(Identity Attribute)을 보유

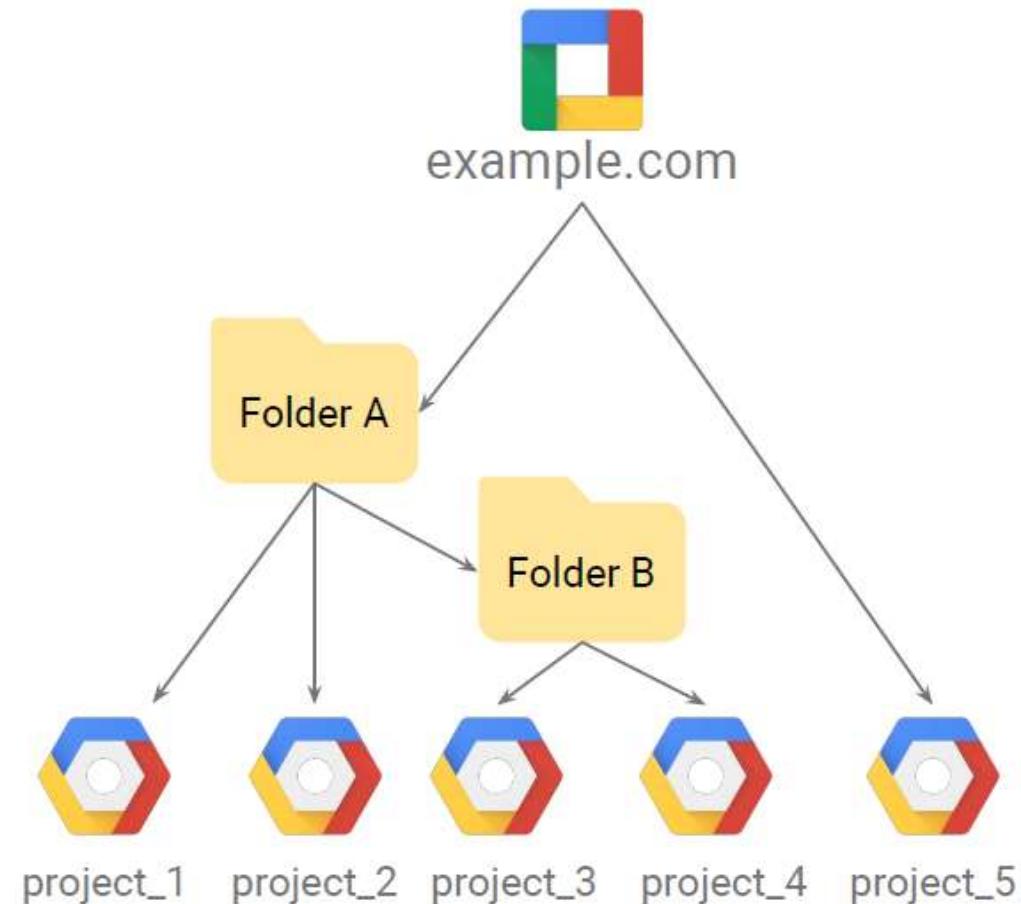
<b>Project ID</b>	Globally unique	Chosen by you	Immutable
<b>Project name</b>	Need not be unique	Chosen by you	Mutable
<b>Project number</b>	Globally unique	Assigned by GCP	Immutable

# 구글클라우드 폴더 (Folder)

## 폴더가 있어서 유연한 관리가 용이

- 조직도에 따른 프로젝트 그룹화 관리 용이
- 폴더는 프로젝트나 하위 폴더 등을 포함할 수 있음
- 폴더를 사용하여 정책(Policy) 설정 및 할당이 가능

→ 대기업 등에서 조직에 따른 권한 및 보안정책 관리 등이 쉽고 편해짐



# 구글클라우드 자원 계층체계와 IAM 역할 (Role)

An example IAM resource hierarchy

- A policy is set on a resource.
- Each policy contains a set of roles and role members.
- Resources inherit policies from parent.
- Resource policies are a union of parent and resource.
- A less restrictive parent policy overrides a more restrictive resource policy.



Who



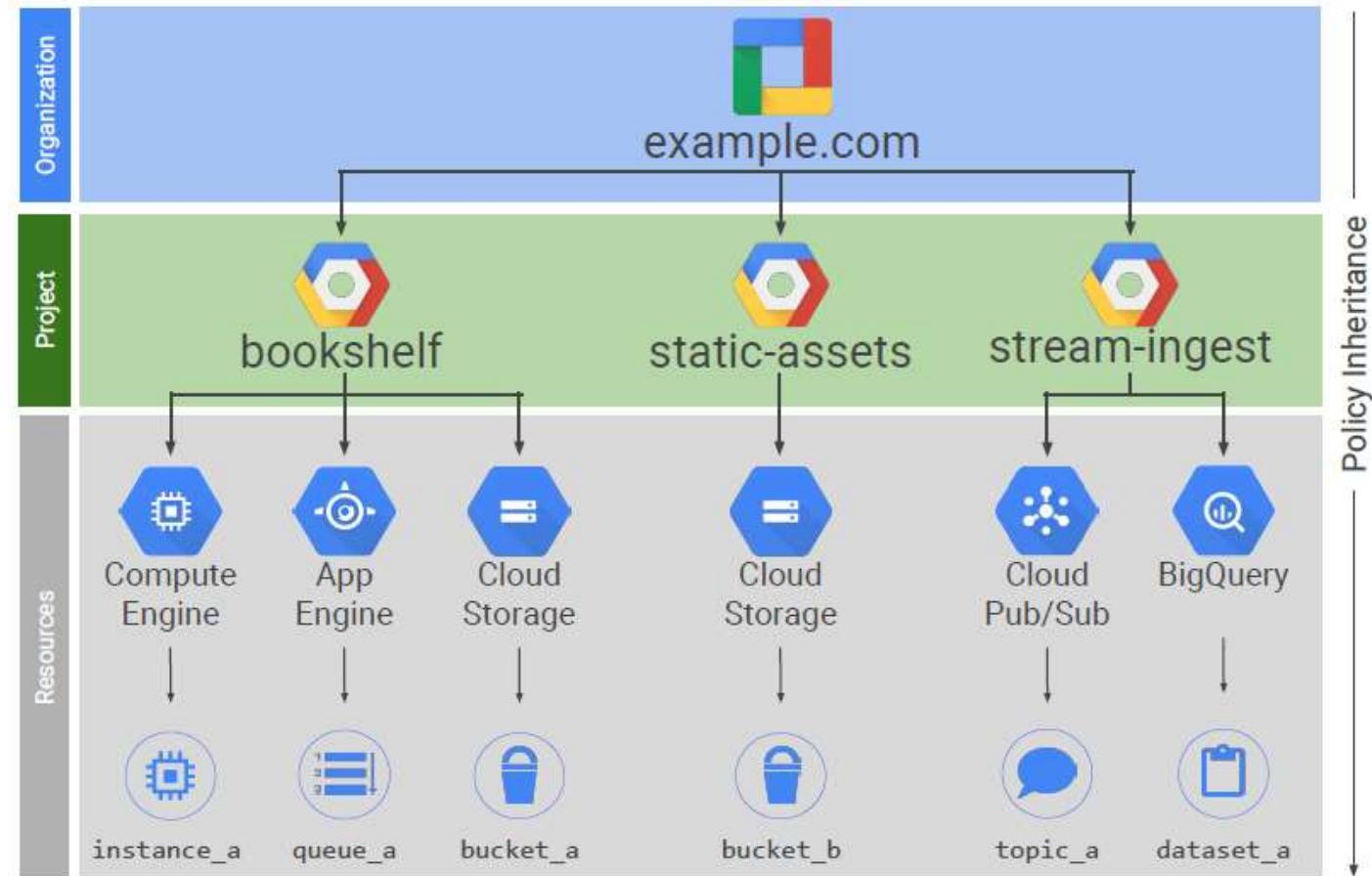
can do what



on which resource

account  
User identity

IAM roles

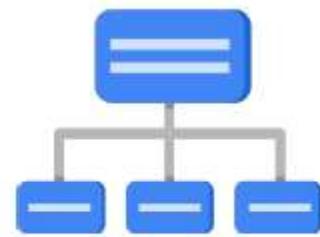


# 구글클라우드 IAM 소개

## Cloud IAM Objects



Cloud IAM



Organization



Folders



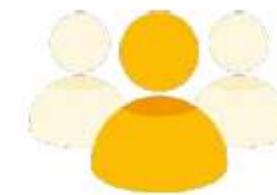
Projects



Resources



Roles



Members

# 구글클라우드 IAM 역할 : 기본 역할 (Primitive Roles)

## IAM Primitive Roles



Owner

- Invite members
- Remove members
- Delete projects
- And...



Editor

- Deploy applications
- Modify code
- Configure services
- And...



Viewer

- Read-only access



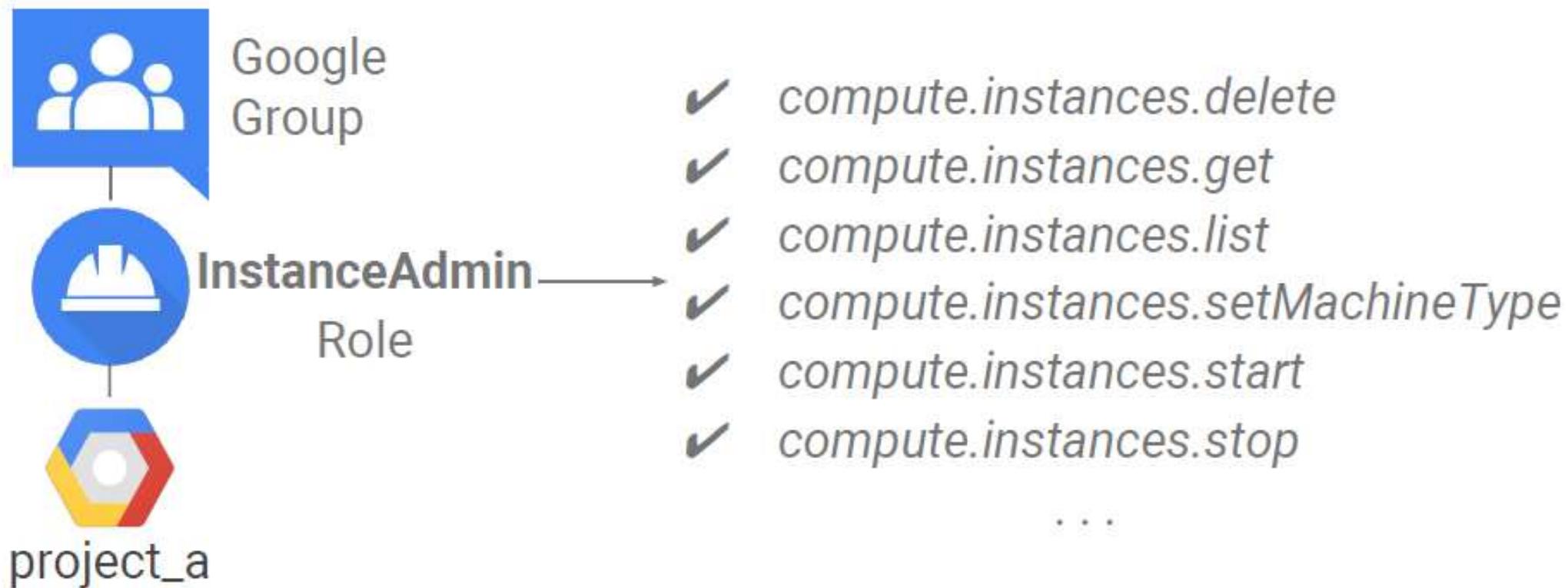
Billing administrator

- Manage billing
- Add and remove administrators

A project can have multiple owners, editors, viewers, and billing administrators.

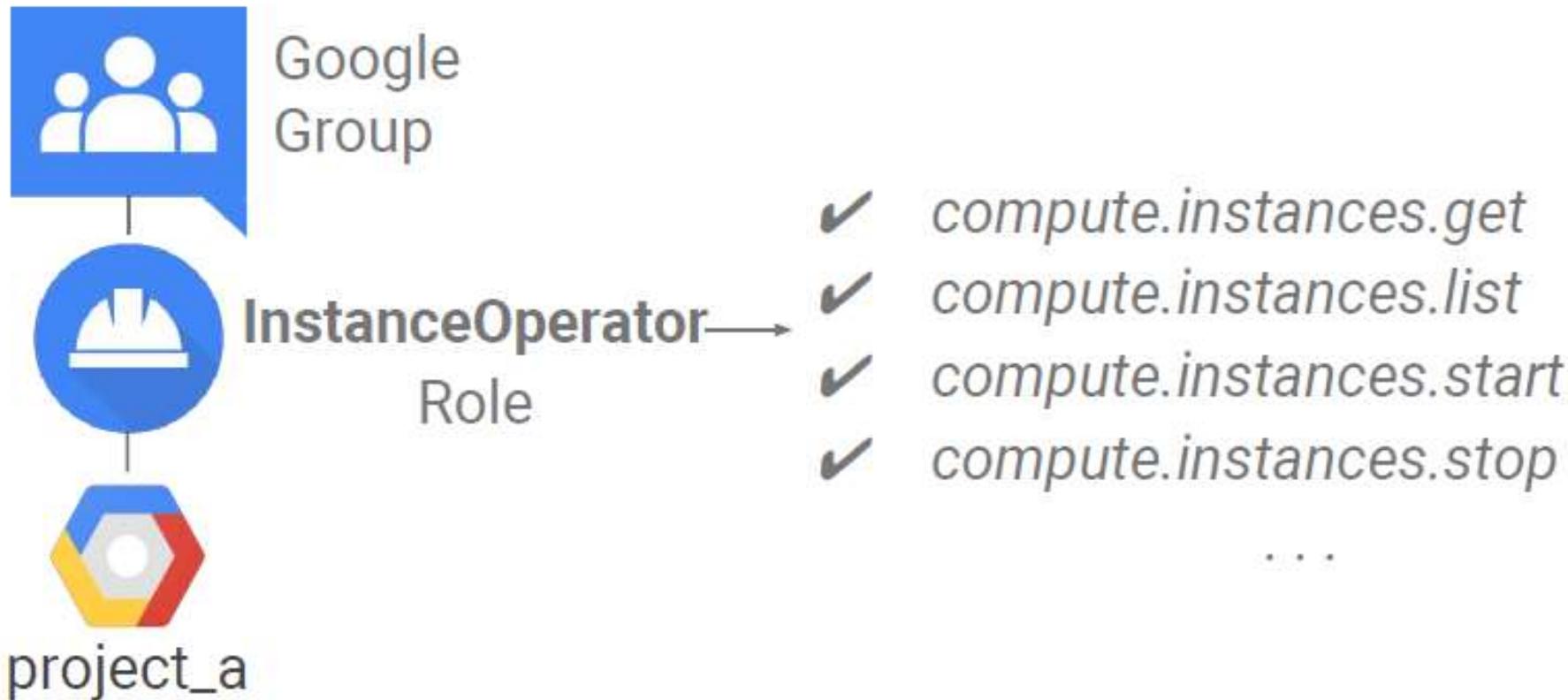
# 구글클라우드 IAM 역할 : 사전 정의된 역할 (Predefined Roles)

- IAM predefined roles offer more fine-grained permissions on particular services



# 구글클라우드 IAM 역할 : 맞춤 역할 (Custom Roles)

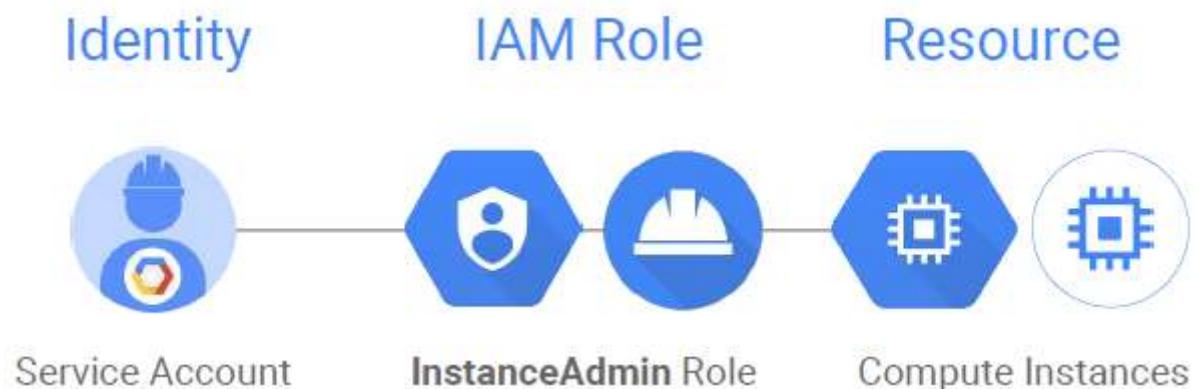
- IAM custom roles let you define a precise set of permissions



# 서비스 계정의 IAM

## 서비스 계정과 IAM

- Service accounts authenticate using keys.
- Google manages keys for Compute Engine and App Engine.
- You can assign a predefined or custom IAM role to the service account.



# 클라우드 아이덴티티

- **Cloud Identity**

Microsoft Active  
Directory or LDAP

Users and groups in  
your existing directory  
service

Google Cloud  
Directory Sync

Scheduled  
one-way sync



Users and groups in  
your Cloud Identity  
domain

# 구글클라우드와 상호작용하는 방법

- Interacting with GCP

**Cloud Platform  
Console**

Web user  
interface



**Cloud Shell and  
Cloud SDK**

Command-line  
interface



**Cloud Console  
Mobile App**

For iOS and  
Android



**REST-based API**

For custom  
applications



## Module 9

# Multi-Cloud Migration & Cost & 운영 전략

## **Part II**

# **Cloud Migration 개요**

# **클라우드 이전**

## **(Cloud Migration)**

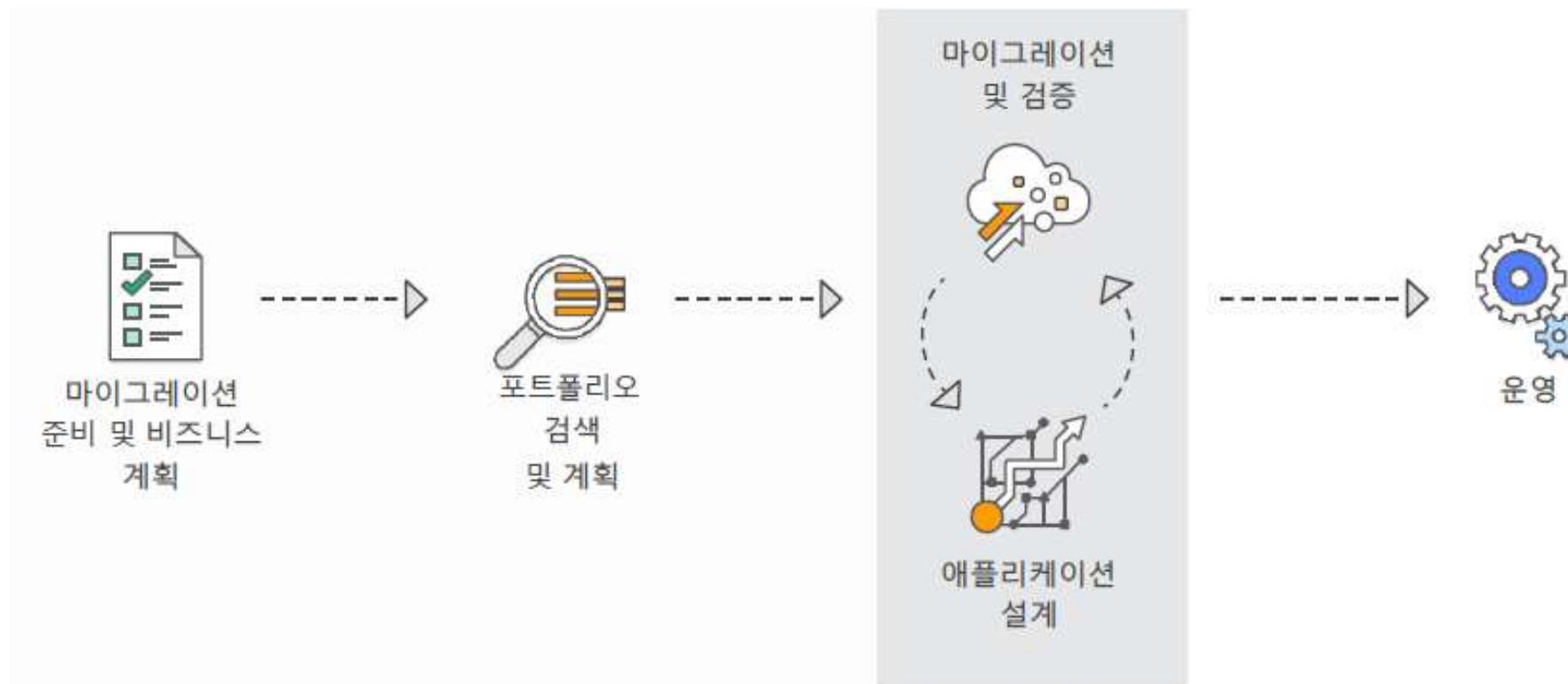
# 클라우드 이전 (Cloud Migration)

- 조직의 기존 IT 자산 중 유의미한 부분을 (전부일 필요는 없음) 클라우드로 이전하는 것
- 두 가지 모델
  - 5 단계 마이그레이션 프로세스
  - 클라우드 마이그레이션을 위한 6 가지 전환 전략 ("6R")



# 클라우드 이전 (Cloud Migration)

- 5 단계 마이그레이션 프로세스



# 클라우드 이전 (Cloud Migration)

## 1 단계:

### マイグ레이션 준비

### 및 비즈니스 계획

계획이 없다는 것은 실패를 계획하는 것



견실한 비즈니스 사례를 개발하기 위해서는 조직의 목표를 기준 애플리케이션의 연식 및 아키텍처, 해당 제약 사항과 함께 고려해야 합니다.

경영진 참여, 빈번한 의사소통, 투명한 목적, 여기에 공격적이면서도 현실적인 목표 및 일정이 더해지면 전체 조직이 마이그레이션 결정을 보다 쉽게 도출할 수 있습니다.

## 2 단계:

### 포트폴리오 검색 및 계획

단기, 중기, 장기

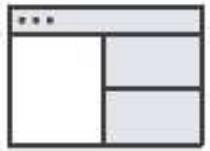


상호 종속성 맵을 포함한 전체 환경포트폴리오 분석, 그리고 마이그레이션 전략 및 우선 순위는 성공적인 마이그레이션 계획 수립을 위한 핵심 요소입니다.

애플리케이션의 복잡성 및 비즈니스 영향은 마이그레이션 방식에 영향을 미칩니다. 포트폴리오 내에서 덜 중요하고 덜 복잡한 애플리케이션부터 마이그레이션 프로세스를 시작하면 팀이 초기 마이그레이션 단계에서 다음과 같은 교훈을 학습할 수 있습니다.

- 초기 학습 단계에서 미션 크리티컬 애플리케이션을 대상으로 연습하는 것이 아니라는 확신
- 향후 반복 작업에 적용할 수 있는 기초 학습
- 역량 및 프로세스 캡을 메우고 경험을 바탕으로 모범 사례를 적극적으로 보강할 수 있는 능력

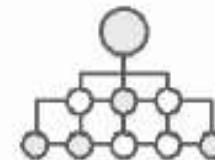
# 클라우드 이전 (Cloud Migration)



**3 단계 / 4 단계:**  
애플리케이션 설계, 마이그레이션 및  
검증  
민첩성, 유연성, 반복성

이 단계들에서는 초점이 포트폴리오에서 개별 애플리케이션으로 옮겨갑니다. 각 애플리케이션이 마이그레이션 전략 ( “6R” ) 중 하나에 따라 설계, 마이그레이션 및 검증됩니다.

대부분의 경우, 연속적 개선 접근 방식이 권장됩니다. 프로젝트 유연성 및 성공 수준은 이 단계들에서 반복적 방법론을 얼마나 잘 적용하는가로 흔히 귀결됩니다.

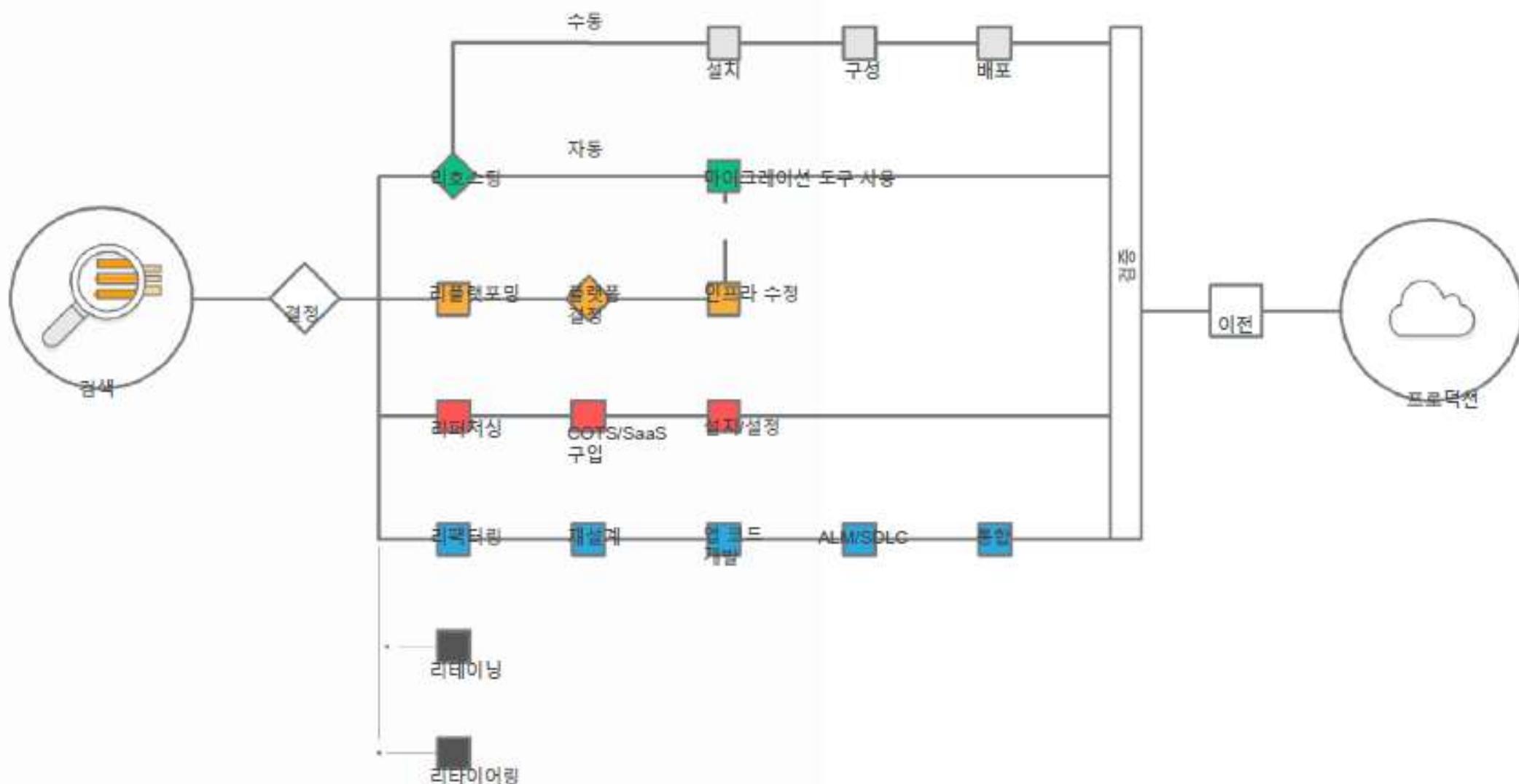


**5 단계:**  
최적화된 운영 모델  
클라우드로 전환

애플리케이션이 마이그레이션됨에 따라 새로운 기반을 최적화하고 기존 시스템을 종료하고 현대적 운영 모델을 목표로 계속 반복하게 됩니다. 운영 모델을 더 많은 애플리케이션을 마이그레이션하면서 지속적으로 개선되는 인력, 프로세스, 기술의 집합으로 생각해 보십시오.

이상적으로는 이미 습득한 기본적 전문 지식을 토대로 운영 모델을 구축하게 될 것입니다. 이것이 여의치 않다면 처음 몇몇 애플리케이션 마이그레이션을 사용하여 이러한 기반을 구축하십시오. 마이그레이션이 가속화됨에 따라 운영 모델이 지속적으로 개선되고 더 정교해질 것입니다.

# 클라우드 이전 (Cloud Migration)



# 클라우드 이전 (Cloud Migration)

## 1. 리호스팅 – “리프트 앤 시프트 (*lift-and-shift*)”

조직이 비즈니스 사례 충족을 위해 마이그레이션을 구현하고 빠르게 확장하고자 하는 대규모 기간계 마이그레이션 시나리오에서는 대부분의 애플리케이션이 리호스팅됩니다.

대부분의 리호스팅은 AWS Server Migration Service(SMS)와 같은 도구를 사용하여 자동화할 수 있지만 일부 고객은 기간계 시스템을 새로운 클라우드 플랫폼에 적용하는 방법을 배우기 위해 수동으로 리호스팅하는 방식을 선호합니다.

또한, 이미 클라우드에서 실행 중인 애플리케이션은 최적화/재설계하는 것이 더욱 쉽다는 사실이 명확해졌습니다. 이는 부분적으로는 조직이 더 나은 최적화/재설계 기술을 개발했기 때문이고 부분적으로는 어려운 부분 (애플리케이션, 데이터, 트래픽 마이그레이션)은 이미 완료되었기 때문입니다.

## 2. 리플랫포밍 – “리프트 킹커 앤

시프트 (*lift-tinker-and-shift*)” 애플리케이션의 핵심 아키텍처는 변경하지 않으면서 일부 실질적인 혜택을 얻기 위해 몇 가지 클라우드 최적화를 실시하는 전략입니다. 예를 들어, 데이터베이스 인스턴스 관리에 소요되는 시간을 단축하기 위해 Amazon Relational Database Service(Amazon RDS) 같은 서비스로서의 데이터베이스 (database-as-a-service) 상품으로 전환할 수 있습니다.

## 3. 리퍼체이싱 – 현재 환경을 변경하기; “드롭 앱 습 (*drop and shop*)”

이 전략은 현재의 솔루션을 새 버전 또는 다른 솔루션으로 전환하는 것으로, 조직이 현재 사용 중인 라이선스 모델을 변경할 의사가 있음을 의미합니다. 새 버전으로 용이하게 업그레이드할 수 있는 워크로드의 경우 이 전략을 통해 기능 집합 업그레이드와 보다 원활한 구현이 가능합니다.

# 클라우드 이전 (Cloud Migration)

**4. 리팩토링/리아키텍팅** – 애플리케이션을 설계 및 개발 방식 변경; 일반적으로 클라우드 최적화 기능을 채택할 때 사용

이 전략은 일반적으로 애플리케이션의 기존 환경에서 달성하기 어려운 기능 추가, 확장 또는 성능 개선에 대한 강력한 비즈니스 요구에 따라 선택됩니다.

조직이 서비스 중심 아키텍처 (SOA)로 전환하여 민첩성을 제고하고 비즈니스 연속성을 개선하고자 할 경우 이 전략은 비록 가장 많은 비용이 소요되는 솔루션이지만 그만큼 채택할 가치가 있습니다.

**5. 리타이어링** – IT 포트폴리오에서 불필요한 부분을 폐기하거나 아카이브 더 이상 유용하지 않아 사용 중지가 가능한 IT 자산을 식별하면 비즈니스 사례에 도움이 될 수 있으며 팀이 폭넓게 사용되는 리소스를 유지관리하는 데 집중할 수 있습니다.

**6. 리테이닝** – 당분간 아무 조치도 하지 않고 추후 검토

IT 포트폴리오에 마이그레이션할 준비가 되지 않은 부분이 있거나 온프레미스에 유지하는 것이 더 안심이 되기 때문에, 또는 애플리케이션을 최근에 업그레이드하여 마이그레이션을 위해 다시 변경할 준비가 되지 않았기 때문에 조직이 IT 포트폴리오의 일부분을 유지하는 경우입니다.

비즈니스에 유의미한 것만을 마이그레이션해야 하지만, 더 많은 포트폴리오를 클라우드로 이전할수록 리테이닝의 이유는 더 적어질 것입니다.

# **Cloud Adoption Framework (CAF)**

*Reference : AWS Cloud Adoption Framework*

*<https://aws.amazon.com/ko/professional-services/CAF/>*

# Cloud Migration

- All change, even a change for the better, is always accompanied by drawbacks and discomforts.

- Arnold Bennett

- Minimalize those drawbacks and discomforts so you can rapidly transform your business with cloud
- Cloud Adoption Framework (CAF) helps organizations understand how cloud adoption transforms the way they work. It leverages our experiences assisting organizations from every segment and every geography with their Cloud Adoption Journey.

Reference : AWS Cloud Adoption Framework

<https://aws.amazon.com/ko/professional-services/CAF/>

# Cloud Adoption Framework(CAF)의 개념

- 클라우드를 받아들이는 과정은 디지털 전환(Digital Transformation)의 과정
- 필요한 역량의 차이(Gap)을 따라잡기 위해 기술을 개발하는 과정에 대한 전사적 기획이 필요

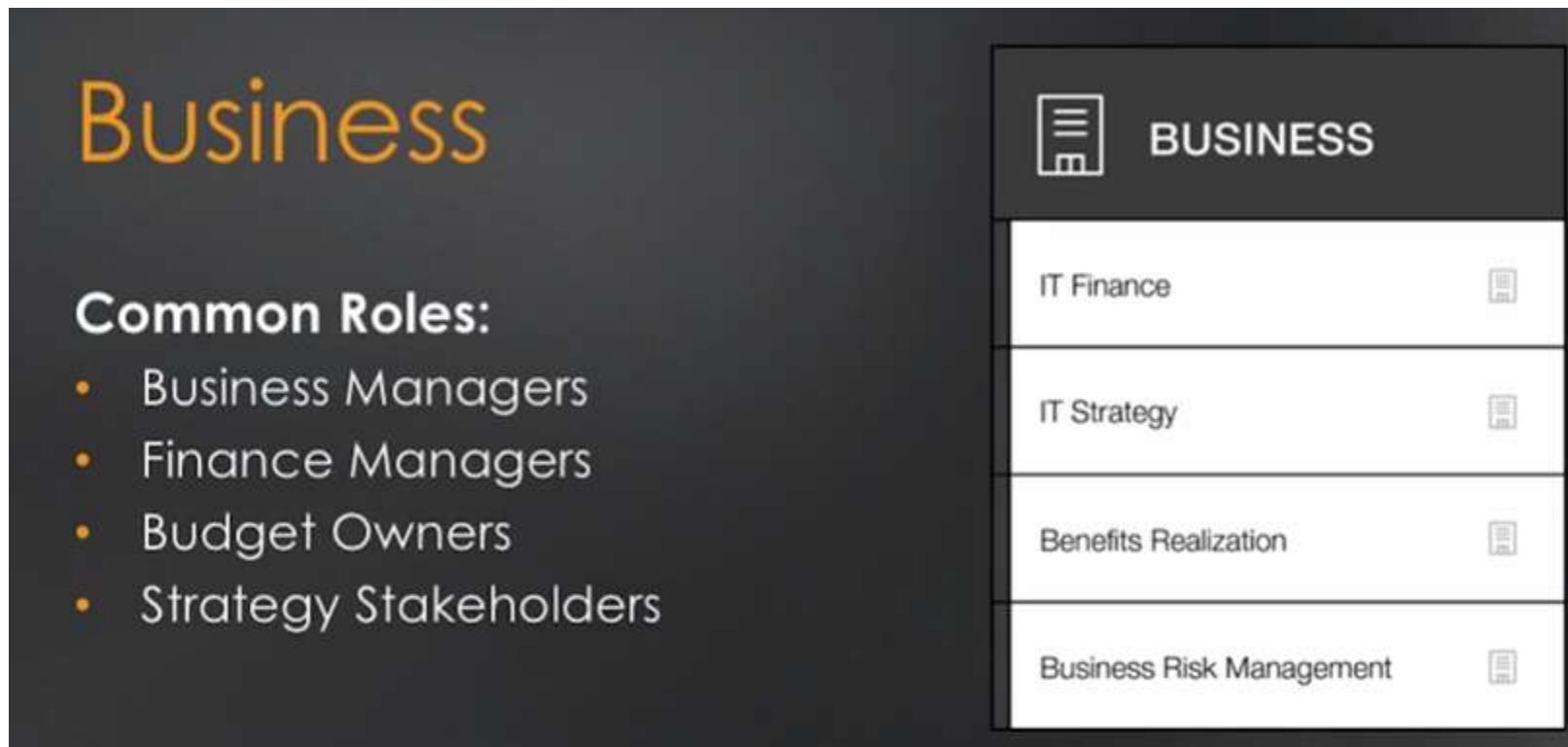


## Digital Transformation

- Capabilities
- Skills
- Processes

# 경영관리 부서의 변화

- 기존 경영진이나 재무/관리부서의 사람들이 해 오던 것과는 다른 관점이 필요하기 때문에, 디지털 전환이나 클라우드의 도입을 전사적으로 계획하는데 있어서 경영관리 분야 사람들의 참여가 필요



# 인사 부서의 변화

- 기술변화의 속도나, Agile DevOps 프로젝트 진행의 속도를 기존 교육이나 인사관리 체계로는 따라가지 못함

People

**Common Roles:**

- Human Resources
- Staffing
- People Managers

PEOPLE

Management Function	User Icon
Resource Management	User icon
Incentive Management	User icon
Career Management	User icon
Training Management	User icon
Organizational Change Management	User icon

# IT 부서의 변화

- 회사의 경영목표에 따라 기존 IT 부서들은 어떻게 운영을 해야 할 것인지 변화가 필요

The diagram illustrates the concept of Governance, showing common roles and specific management functions.

**Governance**

**Common Roles:**

- CIO
- Program Managers
- Project Managers
- Enterprise Architects
- Business Analysts
- Portfolio Managers

**GOVERNANCE**

Portfolio Management	+
Program and Project Management	+
Business Performance Measurement	+
License Management	+

# Platform 기술의 변화

- 플랫폼의 변화가 회사의 Application과 Infrastructure 관점에서 전반적으로 미칠 영향과, 담당자 역할별로 필요한 기술기량 차이(Skill Gap)에 대해 생각을 해야 함



# 보안 기술의 변화

- 기존의 OS Anti-virus나 Firewall 등 네트워크 장비, 암호 등의 관점에서 주로 생각하던 보안이, 역할(Role) 기반의 접근권한(Access Control) 및 컴플라이언스(Compliance) 관리로 바뀌는 것에 따른 전사적인 운영 및 담당자 기술기량의 변화가 필요



- 클라우드 도입으로 인한 주요 성능지표(KPI) 보고서/모니터링, SLA(Service Level Agreement), 버전관리, CI/CD, Disaster Recovery 등에 대한 변화를 고려

# Operations

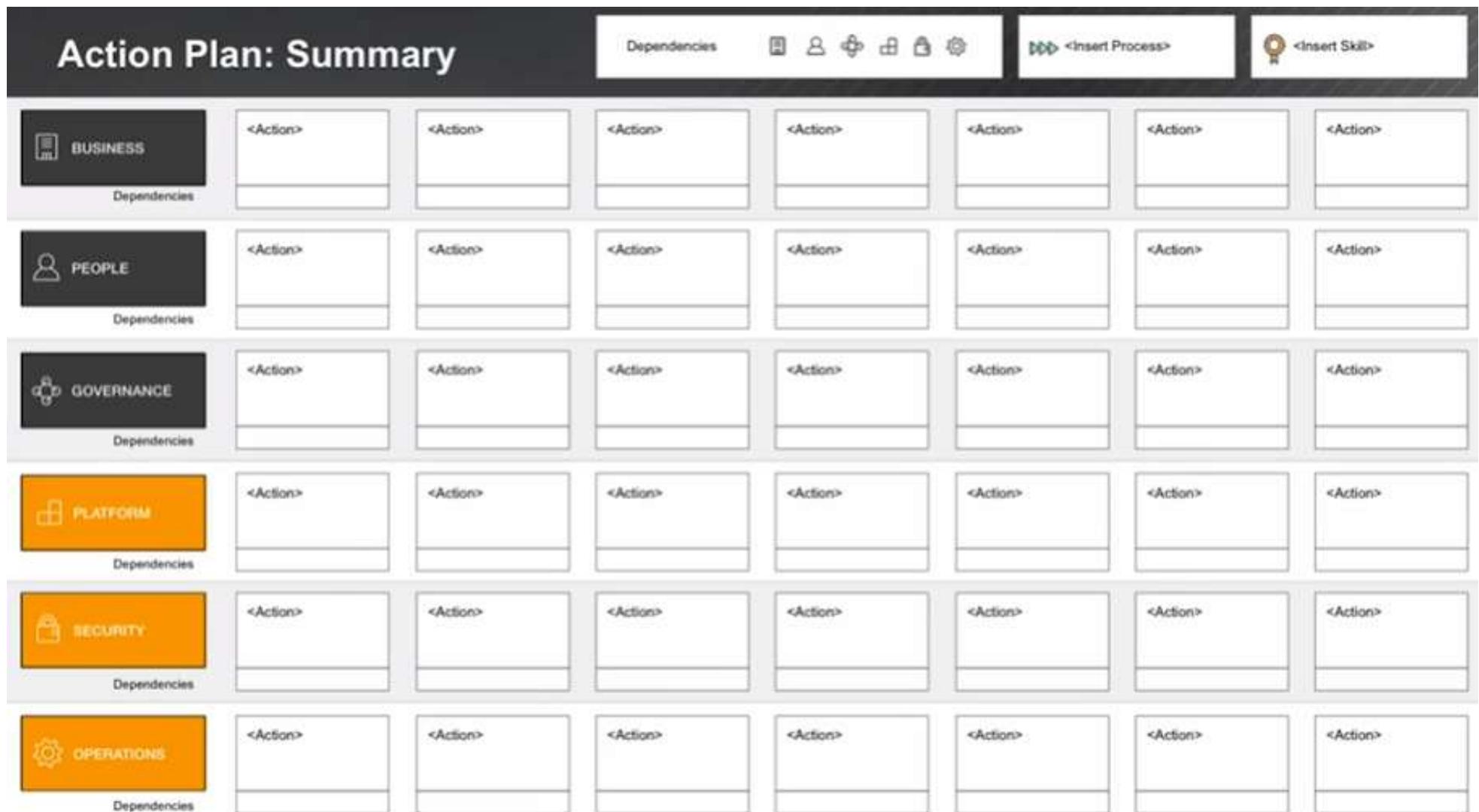
**Common Roles:**

- IT Operations Managers
- IT Support Managers

FUNCTION	ICON
Service Monitoring	gear
Application Performance Monitoring	gear
Resource Inventory Management	gear
Release Management / Change Management	gear
Reporting and Analytics	gear
Business Continuity / Disaster Recovery	gear
IT Service Catalog	gear

# CAF Action Plan 정리 도표 예

- Skill Gap이나 운영방식의 변화에 대한 과정(Process)가 반영되어서 Action Plan을 작성



## The AWS Cloud Adoption Framework:

- Leverages our experiences in assisting organizations from every segment and every geography with their cloud adoption journey.
- Helps your organization to understand how cloud adoption transforms the way the organization works.
- Helps you to identify stakeholders in each perspective that are critical to cloud adoption.
- Enables you to understand cloud adoption from the view of those stakeholders.
- Helps you to create an Action Plan tailored to your organization's needs.

# **Part III**

# **AWS Costing 이해**

# AWS 요금제

# 요금제 적용 방식

- 사용한 만큼 요금을 지불하는 Pay-as-you-go 모델이 기본
- 각 서비스의 요금이 별도 책정 → 필요한 서비스만 선택
- 복잡한 라이선스나 해지 수수료 없이 언제라도 서비스를 시작하나 중단 가능

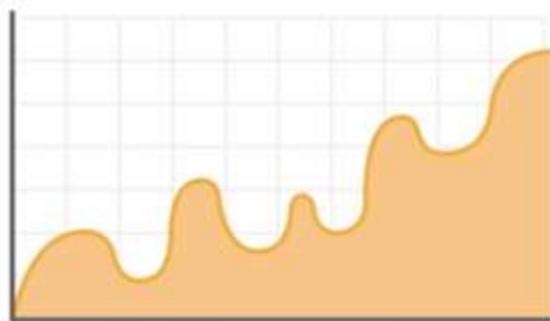


	EC2
	RDS
	ElastiCache
	DynamoDB
	기타 서비스

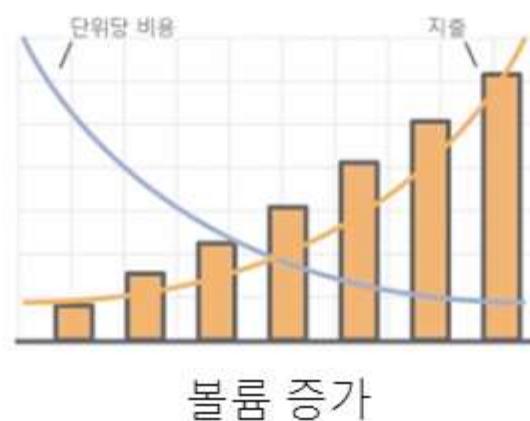
- 유틸리티 기반 모델
- 사용한 만큼만 비용을 지불
- 필요한 서비스만 선택
- 언제든 서비스 시작 또는 중단
- 장기 계약 없음

# 요금제 적용 방식

- 예약 용량에 선수금을 지급할 경우 EC2나 RDS와 같은 서비스에 대해 75%까지 절약 가능



선불형 종량 요금제



볼륨 증가



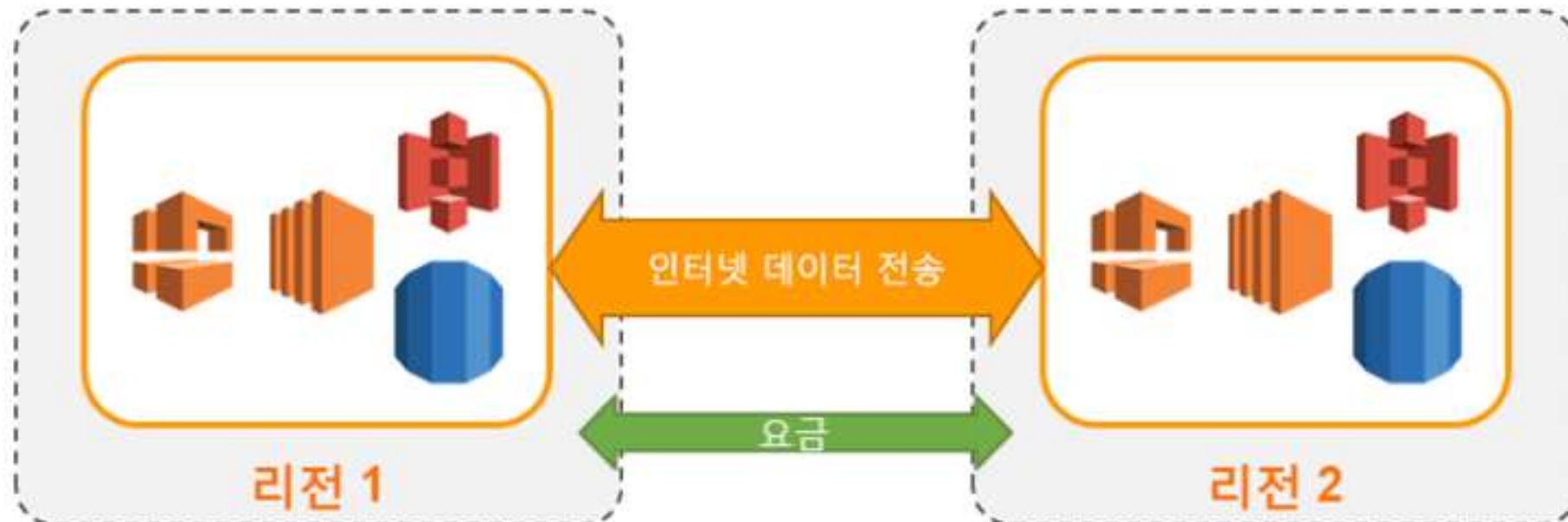
예약 요금제

# 요금 특성

- 동일 리전 내에서 AWS 서비스 간에 전송하는 데이터에는 요금이 발생하지 않음



# 데이터 전송 요금



데이터 전송 요금은 다음 서비스에 적용됩니다.

- Amazon EC2
- Amazon S3
- Amazon RDS
- Amazon DynamoDB
- Amazon SQS
- Amazon SNS
- Amazon VPC

# 월 비용 계산기

## 월 예상 요금 예상:

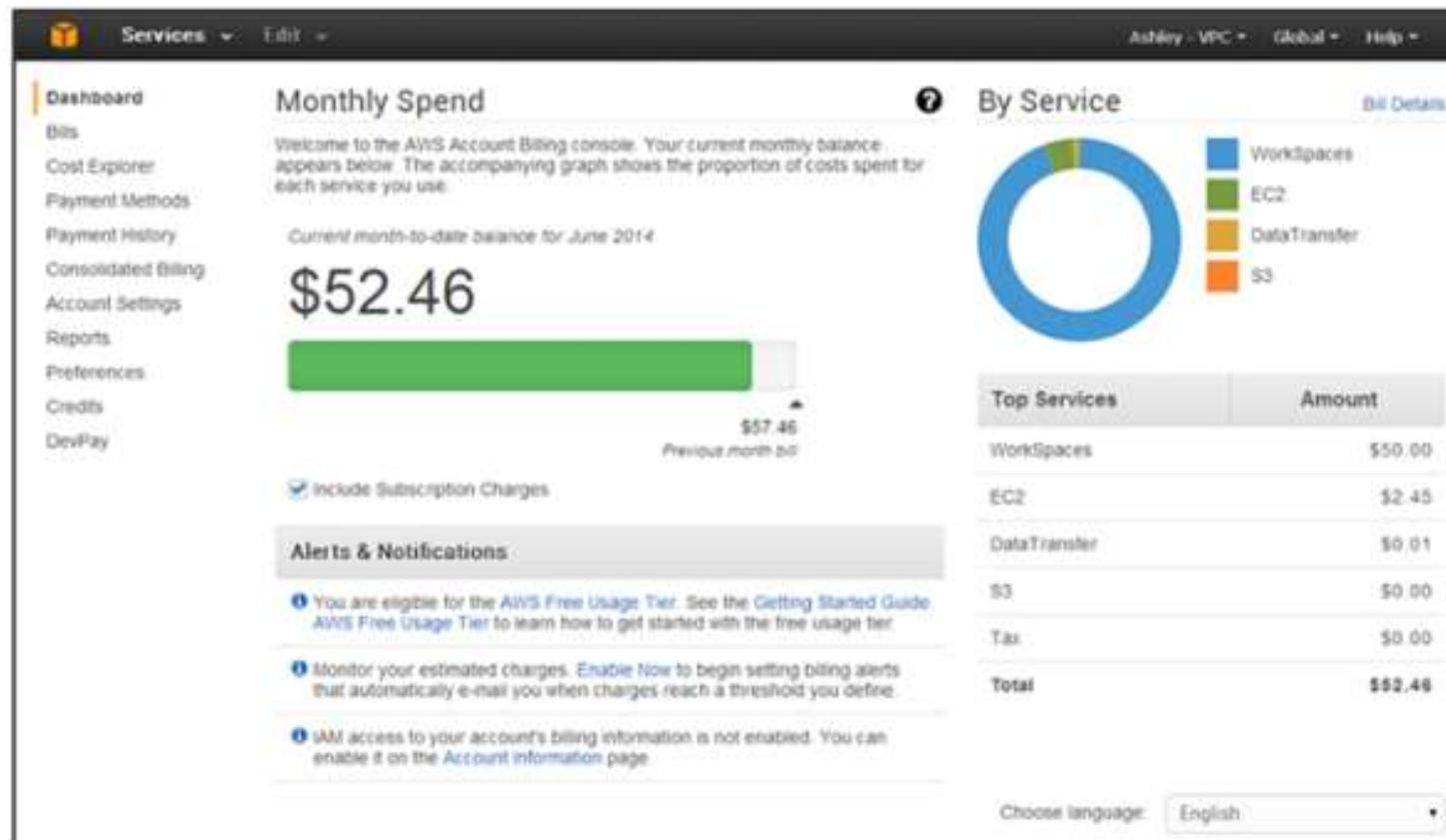
- 서비스당 비용 구분
- 월 예상 요금 합산
- 일반 솔루션을 위한 비용 예상 및 구분

The screenshot shows the AWS Simple Monthly Calculator interface. The top navigation bar includes the Amazon logo, the title "SIMPLE MONTHLY CALCULATOR", and language options (Korean). A sidebar on the left lists various AWS services: Amazon EC2, Amazon S3, Amazon Route 53, Amazon CloudFront, Amazon RDS, and AWS Support. The main content area displays a table for "월별 청구서 금액 (\$ 32.94)".  
Region Selection: "Asia Pacific (Seoul)" is selected from a dropdown menu.  
Service Selection: "Amazon EC2" is selected from a dropdown menu.  
Instance Type: "t2-small" is selected.  
Storage: "Amazon EBS" is selected.  
Volume Type: "SSD" is selected.  
Throughput: "10 GB/s" is selected.  
IOPS: "30" is selected.  
Throughput: "0 Mbps" is selected.  
Cost: "\$ 29.28" is displayed.  
A sidebar on the right lists other regions:

- AWS 무료 빙 서비스
- AWS Elastic Beanstalk 기본 설정
- 아파치 빙 서비스
- 대규모 빙 애플리케이션(모두, 본 디렉토리)
- 미디어 애플리케이션
- 모바일 애플리케이션
- 자체 복구 및 백업

<https://aws.amazon.com/ko/blogs/korea/new-asia-pacific-seoul-region-in-monthly-simple-calculator/>

# 결재 및 비용관리 콘솔



기능:

- 비용 시각화
- 지출 제한에 대한 경고 생성
- 세부 결제 보고서
- 통합 결제
- AWS 계정 통합

# AWS TCO (Total Cost of Ownership) 비용 계산기

# TCO 정의

**TCO** is the initial purchase price of an asset plus its operating costs



Look beyond  
purchase cost

Consider long-term cost

Lower TCO will provide higher  
value over time

## How much does your car really cost?



Price



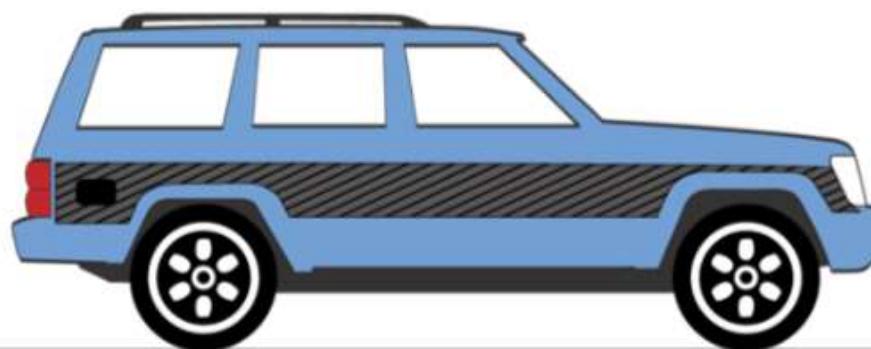
Registration &  
Insurance



Fuel



Maintenance



# TCO와 AWS

Compare the costs of AWS with those of running on-premises or in a colocation facility

Helps you **create a detailed cost comparison**

**Model a specific workload** and compare it

Helps you **communicate value to customers**



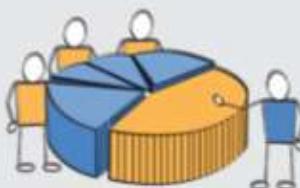
## How Can Customers Lower TCO with AWS?



Replaces up-front capital expense with lower variable cost



Utilizes economies of scale to continually lower costs



Pricing model choice to support variable and stable workloads



Delivers greater savings as they grow

# AWS TCO 분석 요소들



Servers, Network,  
Hardware



Operating System &  
Virtualization Software



Colocation &  
Floor Space



Power & Cooling



Data Center Personnel



Storage Redundancy



**OpsWorks**

Resource Management  
& Software Automation



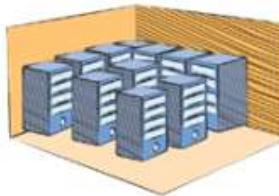
Software-Defined  
Networking

## A Conceptual Representation of TCO



On-Premises

With on-premises solutions, the customer must buy everything



Colocation

The customer must purchase the OS, virtualization, and management software. Floor space, power, and cooling are bundled together



AWS

AWS takes care of everything

Servers

Storage

Network

Software

Staffing

Floor space

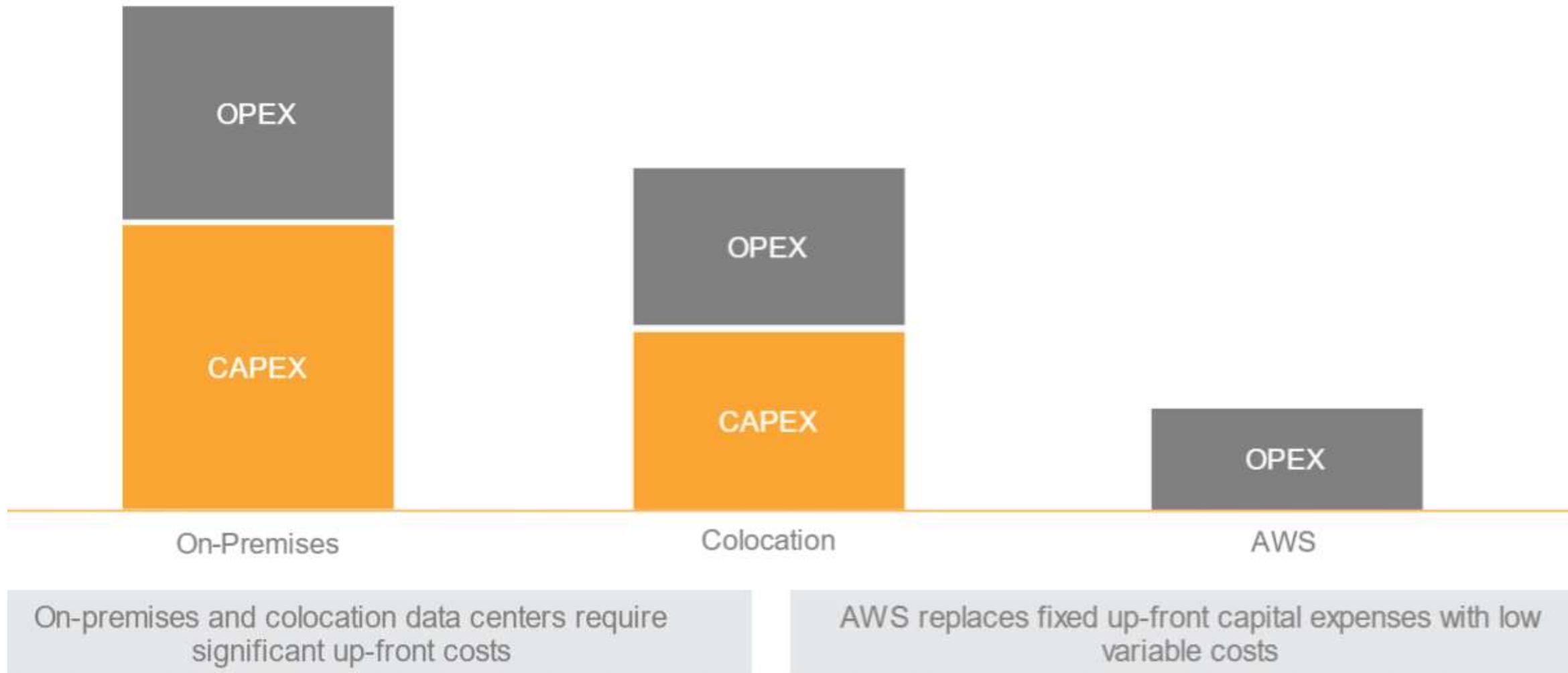
Utilities

Durability

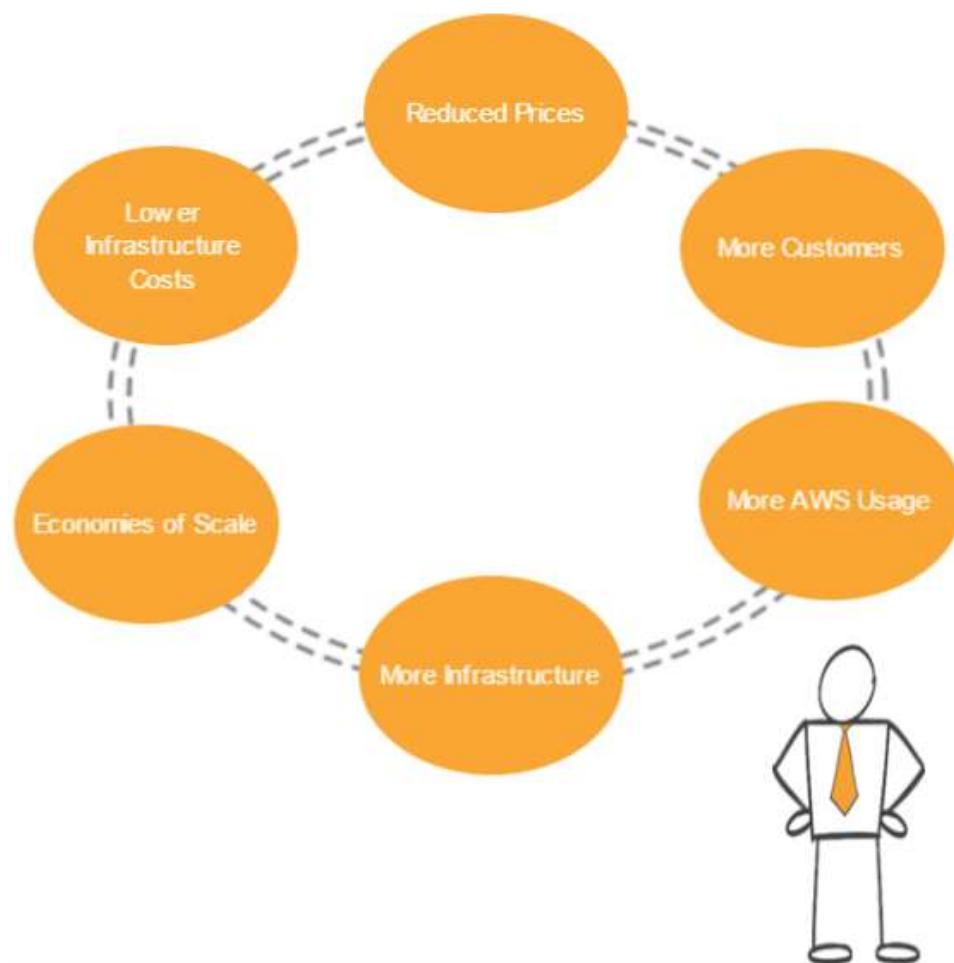
Availability

Physical security

# IT CAPEX Cost-Down



## AWS Pricing Philosophy



- The more that customers use the platform, the more infrastructure AWS has to purchase
- This results in economies of scale
- These cost savings are passed on to customers
- AWS has reduced AWS prices numerous times

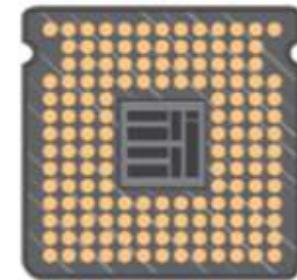
# 가격 핵심 요소



Data transfer



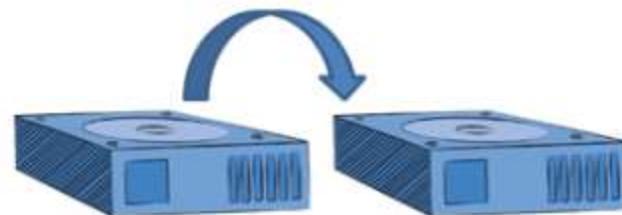
Storage



Compute

These are the fundamental core characteristics that have the **greatest influence** on the **costs** that customers will see

## 가격 핵심 요소



### Data transfer

Data transfer **in** = free

Data transfer **out** = tiered pricing

**Global solution**, certain charges will vary depending on the location's **market, taxes, and utility costs**



### Storage

Pricing dependent on a customer's need

**Amazon S3** customers may receive discounts based on their usage, and they pay only for the storage they use, with no minimum fee

## Compute



-  On-Demand
-  Reserved
-  Spot
-  Dedicated

# AWS TCO 계산기

- 온프레미스와 AWS 비교
- 온프레미스 또는 구성 설명

[www.awstcoccalculator.com](http://www.awstcoccalculator.com)

The screenshot shows the AWS TCO Calculator interface. At the top, it asks for 'Select Currency' (United States Dollar) and 'What type of environment are you comparing against?' (On-Premises). It also specifies the 'AWS region' as 'US East (N. Virginia)' and the 'AWS service' as 'Virtual Machines'. Under 'Servers', there's a table for configuration details with columns: App. Name, Number of VMs, CPU Cores, Memory(GB), Hypervisor, and Guest OS. A dropdown for 'Hypervisor' shows 'VMware' selected. Under 'Storage', there's a table for storage footprint details with columns: Storage Type, Raw Storage Capacity, and a dropdown for 'Storage Type' showing 'SAN'. A 'Calculate TCO' button is at the bottom right.

[https://docs.aws.amazon.com/ko\\_kr/pricing-calculator/latest/userguide/getting-started.html](https://docs.aws.amazon.com/ko_kr/pricing-calculator/latest/userguide/getting-started.html)  
<https://aws.amazon.com/ko/tco-calculator/>

# **GCP 관리**

# GCP 클라우드 자원 관리

- GCP는 자원관리 가시성이 뛰어남

The screenshot shows the left sidebar of the Google Cloud Platform interface. At the top is the 'Google Cloud Platform' logo. Below it are several quick links: Home, BigQuery, Dataflow, Marketplace, Billing, APIs & Services, Support, IAM & admin, Getting started, and Security. Under the 'PRODUCTS' heading, there are more links: Marketplace, Billing, APIs & Services, Support, IAM & admin, Getting started, and Security. The 'IAM & admin' section is expanded, showing sub-links: IAM, Identity & Organization, Policy Troubleshooter, Organization policies, Quotas, Service accounts, Labels, Settings, Privacy & Security, Cryptographic keys, Identity-Aware Proxy, Roles, Audit Logs, and Manage resources. The 'Manage resources' link is highlighted with a light gray background.

The screenshot shows the 'Manage resources' page. At the top, there are buttons for 'CREATE PROJECT' and 'DELETE'. Below is a 'Filter tree' button. A table lists four projects: 'No organization' (ID 0), 'fintech-info' (ID fintech-info), 'ml-test-191114' (ID ml-test-191114), and 'test20191004' (ID test20191004). Each project row has a checkbox and three vertical dots for actions. At the bottom, a message says '0 RESOURCES PENDING DELETION'.

	Name	ID	Status	Charges	Labels	Actions
<input type="checkbox"/>	▼ No organization	0				⋮
<input type="checkbox"/>	• fintech-info	fintech-info				⋮
<input type="checkbox"/>	• ml-test-191114	ml-test-191114				⋮
<input type="checkbox"/>	• test20191004	test20191004				⋮

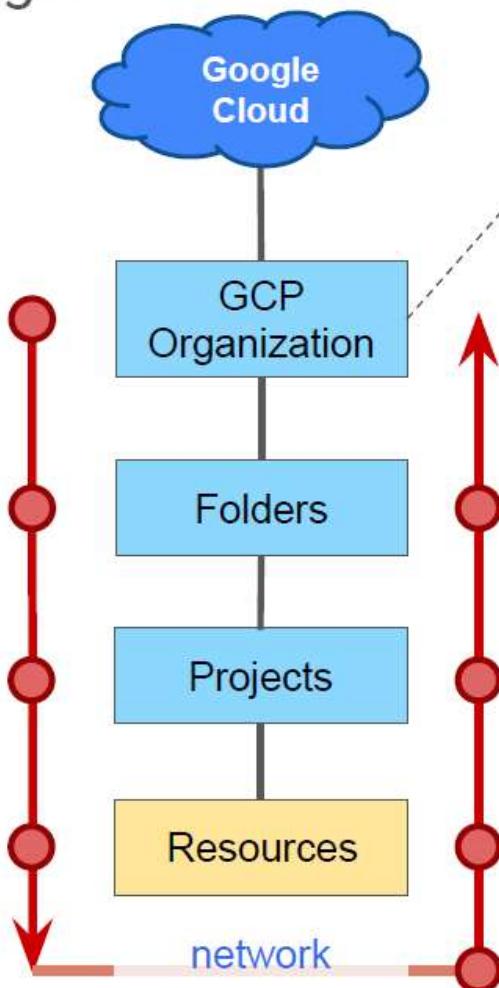
# GCP 클라우드 자원 관리

- 조직 / IAM / 자원관리 / 비용청구 등이 모두 일관성 있게 연계되어 있음

## Cloud Resource Manager

### Identity and Access Management

- Policies are set on resources
  - Roles
  - Members
- Resources inherit policies from parent
- Resource policies are a union of parent and resource
- If parent policy is less restrictive, it overrides the more restrictive resource policy



### Billing and Resource Monitoring

- Organization contains all billing accounts
- Project is associated with one billing account
- Project accumulates consumption of all resources
- A resource belongs to one and only one project
- Resource consumption is measured on:
  - Rate of use/time
  - Number of items
  - Feature use

# GCP 클라우드 Quota

Google Cloud Platform fintech-info

Quotas + EDIT QUOTAS

Quota type	Service	Metric	Location	Current Usage	7 Day Peak Usage	Limit
All quotas	All services	All metrics	All locations	15.519	171	2,000
<input type="checkbox"/> Service		Location				
<input type="checkbox"/> Compute Engine API	Global	15.519	171	2,000		
List requests per 100 seconds						
<input type="checkbox"/> Cloud Runtime Configuration API	Global	0 B	446 B	4,194,304 B (4.194 MB)		
Config Storage Usage						
<input type="checkbox"/> Stackdriver Logging API	Global	0.001	1	60,000		
Log ingestion requests per minute						
<input type="checkbox"/> Compute Engine API	Global	0.006	0.006	2,000		
Queries per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	0.005	0.005	2,000		
Read requests per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	3	3	Unlimited		
Queries per day						
<input type="checkbox"/> Compute Engine API	Global	0	0	2,000		
Operation read requests per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	0	0	1,000		
Heavy-weight read requests per 100 seconds						
<input type="checkbox"/> Compute Engine API	Global	0	0	1,000		
Heavy-weight mutation requests per 100 seconds						

# GCP 클라우드 예산 및 알람 설정

Google Cloud Platform ≡ 🔍 ▼

Billing ← Create Budget

- Overview
- Reports
- Cost table
- Cost breakdown
- Commitments
- Budgets & alerts**
- Billing export
- Transactions
- Payment settings
- Payment method
- Account management

1 Scope — 2 Amount — 3 Actions

A budget enables you to track your actual spend against your planned spend.

Name \*

A budget can be scoped to focus on a specific set of resources.

Projects

Products

**NEXT** **CANCEL**

# GCP 클라우드 비용계산기

The screenshot shows the Google Cloud Platform Pricing Calculator. At the top, there is a navigation bar with the Google Cloud logo, a search icon, and language selection (Language: ko). Below the navigation bar, the title "Google Cloud Platform Pricing Calculator" is displayed, along with a note that prices are up to date as of January 31, 2020. A horizontal menu bar below the title contains icons for various services: COMPUTE ENGINE, APP ENGINE, KUBERNETES ENGINE, CLOUD RUN, CLOUD STORAGE, NETWORKING, BIGQUERY, BIGQUERY ML, and DATA. The "COMPUTE ENGINE" icon is highlighted with a red underline. To the right of the menu is a large blue button labeled "Estimate". Below the menu, there is a search bar with the placeholder text "Search for a product you are interested in," followed by several dropdown menus for configuring instances. These dropdowns include "Number of instances" (set to 1), "Operating System / Software" (set to "Free: Debian, CentOS, CoreOS, Ubuntu, or other User Provided OS"), "Machine Class" (set to "Regular"), "Machine Family" (set to "General purpose"), and "Series" (set to "N1"). Each dropdown has a question mark icon to its right.

<https://cloud.google.com/sql/docs/mysql/create-instance?hl=ko>

클라우드 컴퓨팅 기초 교육 (Co-Education Course)

아마존 & 구글 클라우드 기본 원리와 실습

## Day 3 – Module 6

# AWS vs. GCP 비교 및 멀티클라우드 기본 개념 이해

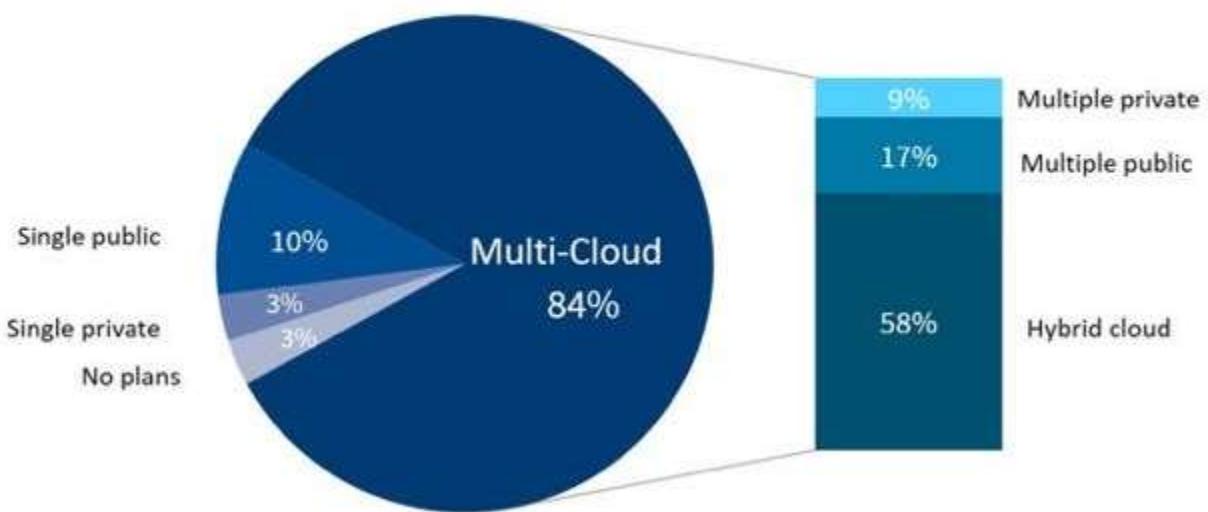
## **Part II**

# **하이브리드/멀티클라우드 기본 개념 이해**

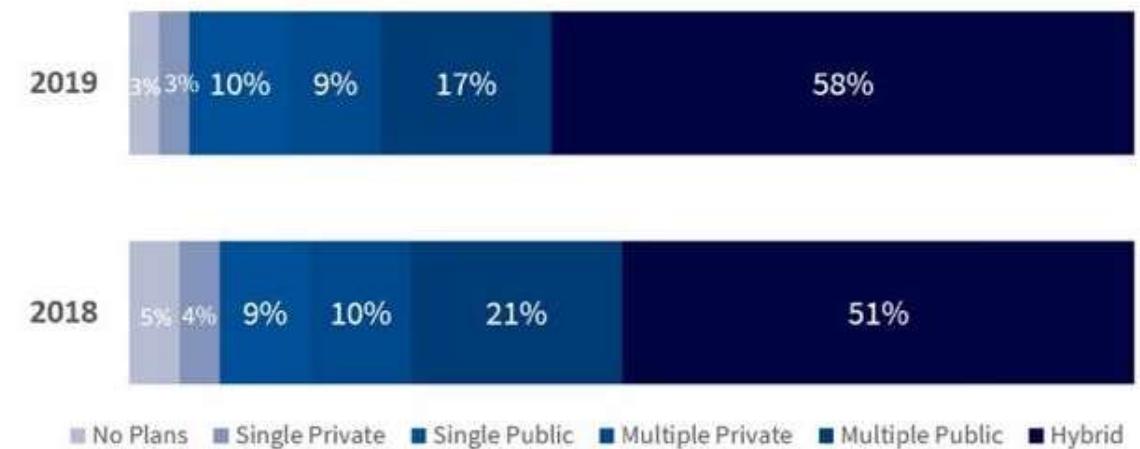
# 대기업 클라우드 전략 비교

## Enterprise Cloud Strategy

1000+ Employees



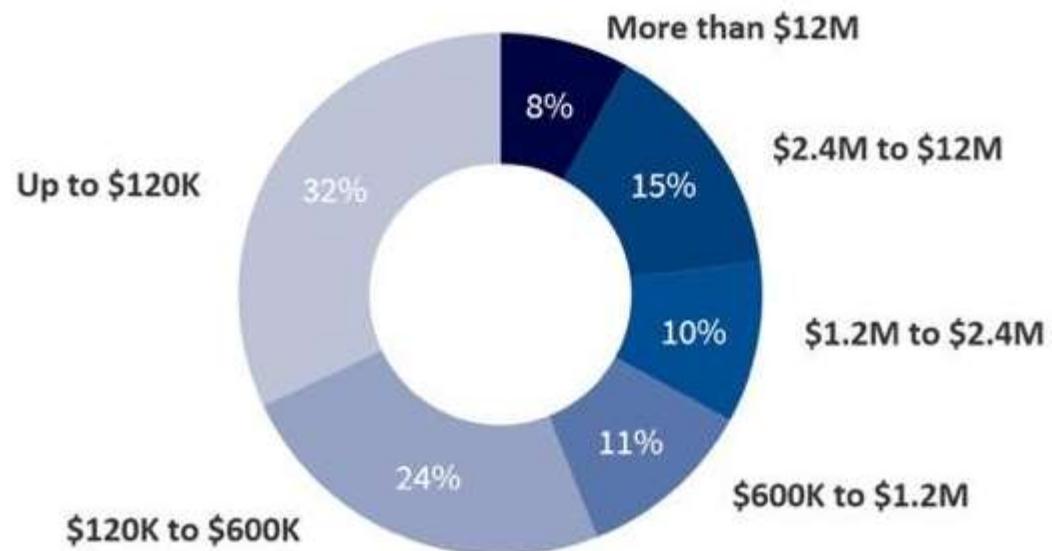
## Enterprise Multi-Cloud Strategy YoY



# 대기업 클라우드 예산 및 사용량 비교

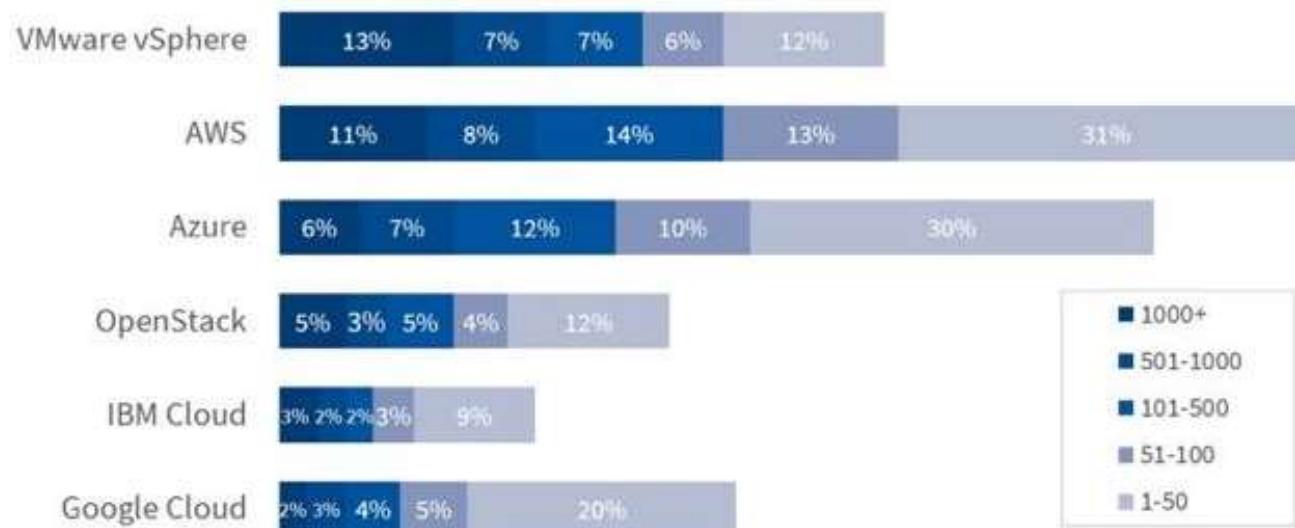
## Annual Public Cloud Spend

All respondents



## # of VMs in Clouds

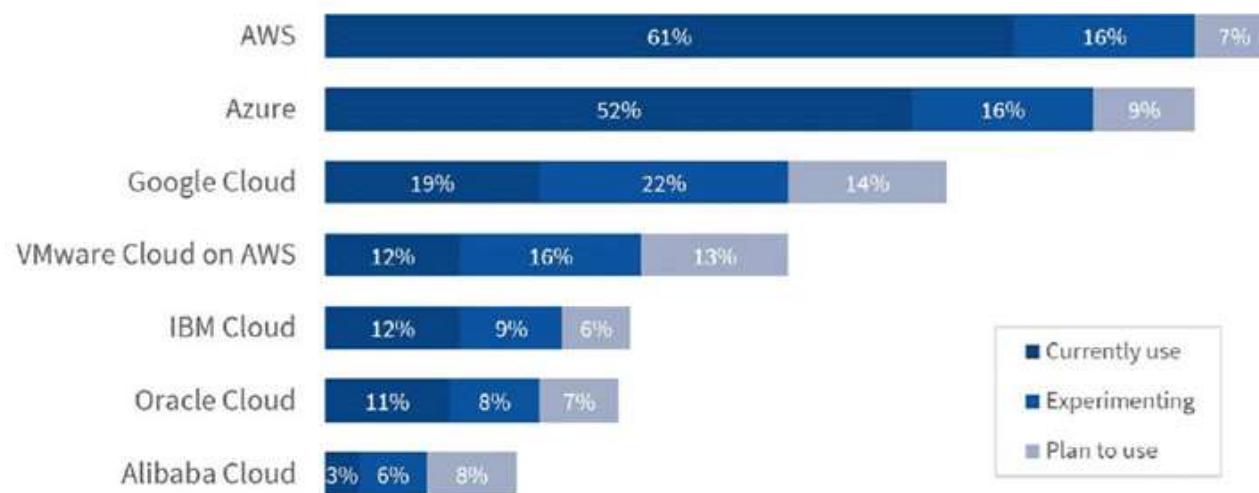
% of All Respondents



# 퍼블릭 클라우드 업체 비교: Public Cloud Adoption

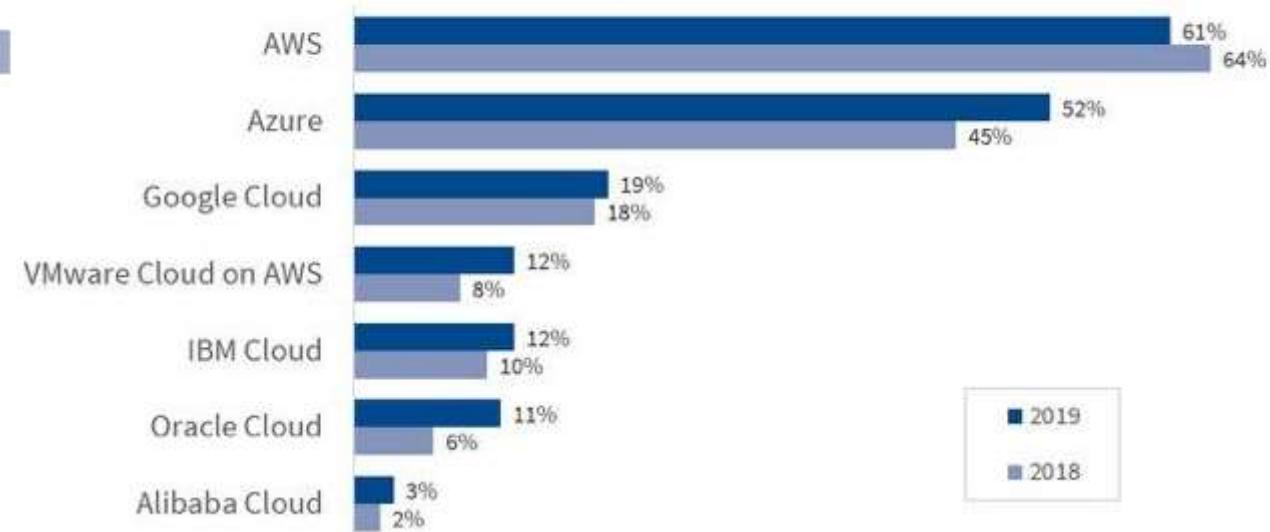
## Public Cloud Adoption

% of All Respondents



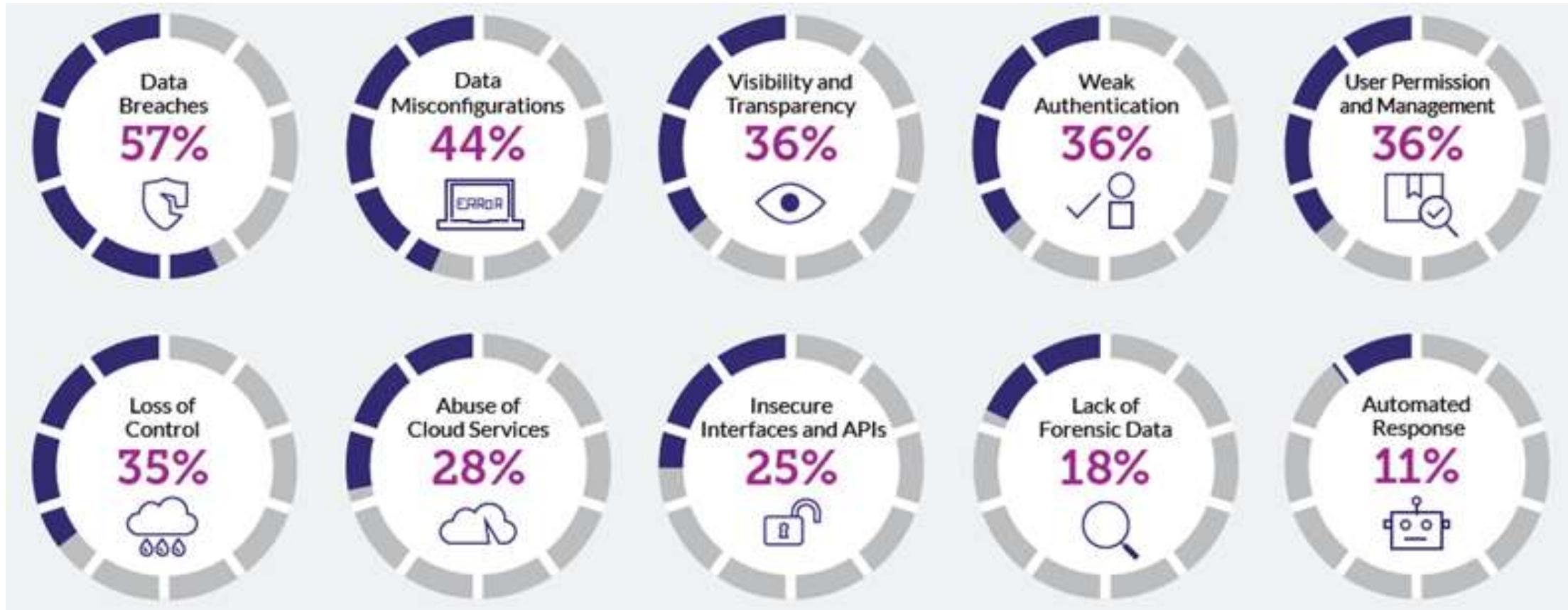
## Public Cloud Adoption

% of All Respondents



# 대기업 클라우드 고민사항

## 클라우드 전문 인력 및 경험 부족



DivvyCloud “State of Enterprise Cloud and Container Adoption and Security” Report (2019)

# 하이브리드/멀티클라우드 – 필요성

The screenshot shows the Chosun.com website interface. At the top, there's a navigation bar with links for 'chosun.com', 'Chosun', 'MICRO SOFTWARE', '로그인' (Login), and '회원가입' (Sign Up). Below the navigation is a search bar and social media sharing icons for Facebook, Twitter, and Email. The main menu includes categories like '#속보', '#기업' (highlighted in red), '#기술', '#자동차', and '#게임·라이프'. On the left, there's a sidebar for '기업' (Business) news, specifically '컴퓨팅 IT 서비스'. The main content area features a large headline: 'AWS 대규모 접속장애...' (Large-scale connection outage...) followed by '단일 클라우드' (Single cloud) 사용한 고객사 피해 커(종합)' (Customer damage increases due to single cloud usage). Below the headline are author details: '최용석 기자 차현아 기자 오세웅 기자'. To the right of the headline are icons for bookmarking, email, and printing. The article summary starts with: '전 세계 1위 퍼블릭 클라우드인 아마존웹서비스(AWS)에서 22일 오전 접속 장애가 발생, AWS에 기반을 둔 국내 IT 서비스가 동시에 마비됐다.' (A major connection outage occurred at Amazon Web Services (AWS) on the morning of the 22nd, causing simultaneous paralysis of IT services based on AWS). The text continues to describe the specific circumstances of the outage.

전 세계 1위 퍼블릭 클라우드인 아마존웹서비스(AWS)에서 22일 오전 접속 장애가 발생, AWS에 기반을 둔 국내 IT 서비스가 동시에 마비됐다.

이번 AWS 접속 장애는 전 세계적인 상황이 아닌 국내에서만 발생한 것으로 나타났다. 오전 8시경(관련기사)부터 AWS의 주요 서비스인 EC2(Elastic Compute Cloud)의 서울 리전(Region, 아마존 지역 데이터센터 명칭)에서 내부 DNS(Domain Name System) 변환에 실패해 외부 접속이 불가능한 DNS 오류가 발생했으며, 이후 아마존의 다른 클라우드 서비스로 장애가 확산된 것으로 드러났다.

특히 이번 장애로 AWS 서울 리전만 단독으로 이용하는 기업들의 피해가 큰 것으로 나타났다. 서울 리전 외에 일본, 싱가포르 등 다른 지역 리전을 동시에 이용하는 멀티클라우드를 적용한 기업들의 경우 AWS를 이용함에도 장애를 겪지 않은 것으로 나타났다.

## 하이브리드 클라우드 – 핀테크 등 규제 산업

- 대부분 애플리케이션은 인증, 결제, 물류 등을 위해 하나 이상의 외부 API 기반 서비스를 이용하기 때문에 하이브리드
- 내부 관리 앱이 인증을 위해 애저 액티브 디렉토리 또는 옥타(Okta)를 호출하는 경우 실제로는 하이브리드 클라우드를 운용하고 있는 것
- 웹사이트에 페이팔(Paypal) 또는 비자(Visa) 결제 위젯이 있다면, 하이브리드 클라우드를 이용하고 있는 것

# 하이브리드/멀티클라우드 – 장단점

## 장점

- 클라우드 옵션 증가
- 특정 서비스에 종속되지 않음
- 유연성 확보
- 클라우드 비용 절감
- 더 쉽고 빠른 재난 복구

## 단점

- 기술 인력 필요량 증가
- 복잡성 증가
- 관리비용 증가

# 하이브리드/멀티클라우드 – 도입 구축

## ● 도입 구축 팁 (Tip)

- 전략적 파트너 (서비스 관리에 업체 서비스/솔루션) 활용
- 현업을 교육
- 애플리케이션 워크로드(Work Load) 클라우드 정책
- 애플리케이션과 데이터 소스 간의 통합 데브옵스 (DevOps)
- 벤더 록인(Lock-in) 문제를 진지하게 고려하고 정책을 설정
- 리소스 태깅(Tagging) 정책
- CMP(Cloud Management Platform) 구축
- 오케스트레이션(Orchestration) 시스템을 사용
- 필수 보안 및 컴플라이언스에 주의
- API를 활용
- 런타임(Run-time) 의존성을 피하고 3<sup>rd</sup> party 솔루션의 복잡성을 해결

# 하이브리드/멀티클라우드 – 복잡성

- 마이크로서비스 API의 복잡성

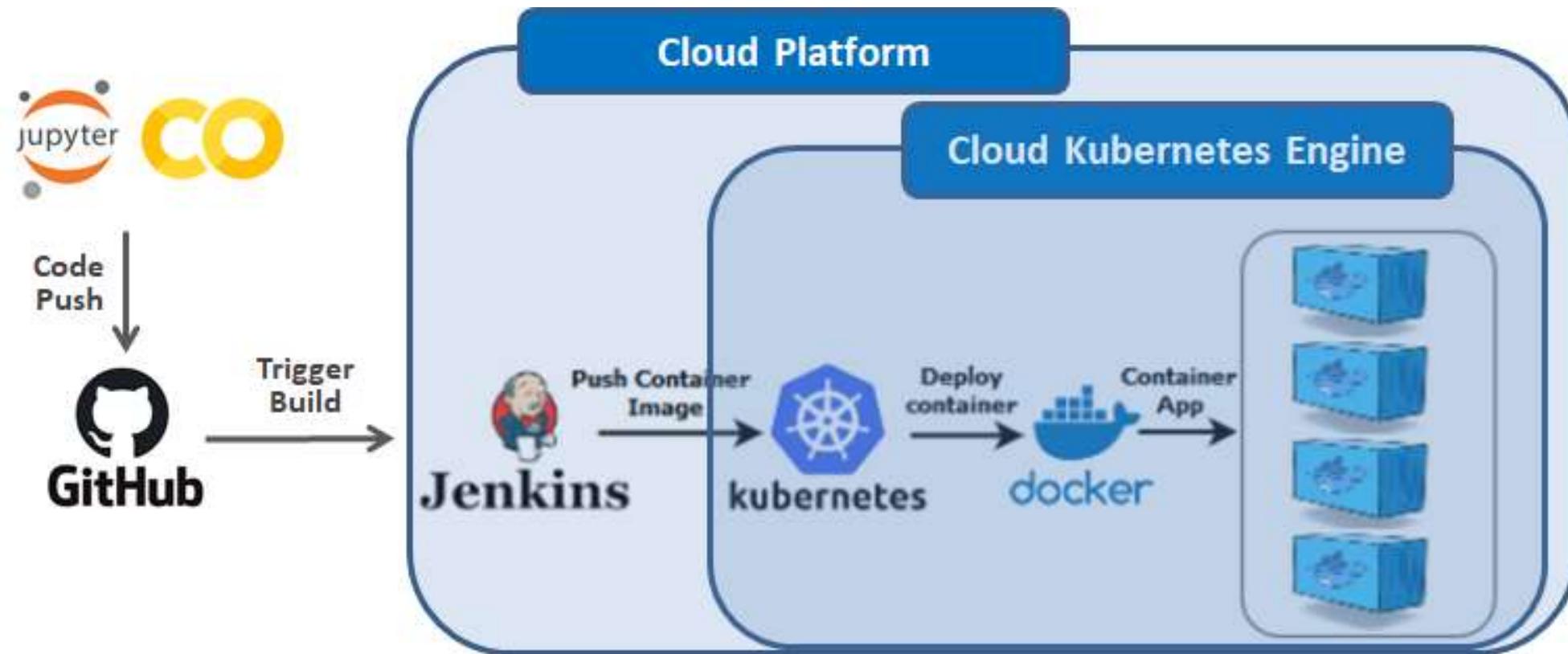
## AWS 정전 사례 (2018.3.2. 미국 동부)

인프라 관점에서 AWS는 사소한 정전이 발생했고 시스템이 꽤 짧은 시간 안에 복구되었다. 하지만 백엔드 흐름을 위해 AWS DC(Direct Connect)에 의존했던 애플리케이션은 초기 사건 후 몇 시간 동안 계속 문제를 일으켰다. 아틀라시안(Atlassian), 슬랙(Slack), 트윌리오(Twilio) 등 여러 애플리케이션 서비스 업체가 여러 클라우드들 사이의 숨겨진 의존성을 고려하지 못했다.

싸우전드아이즈 (ThousandEyes)는 240개 이상의 주요 서비스가 정전의 영향을 받았다고 밝혔다.

# 하이브리드/멀티클라우드 운영 – CI/CD

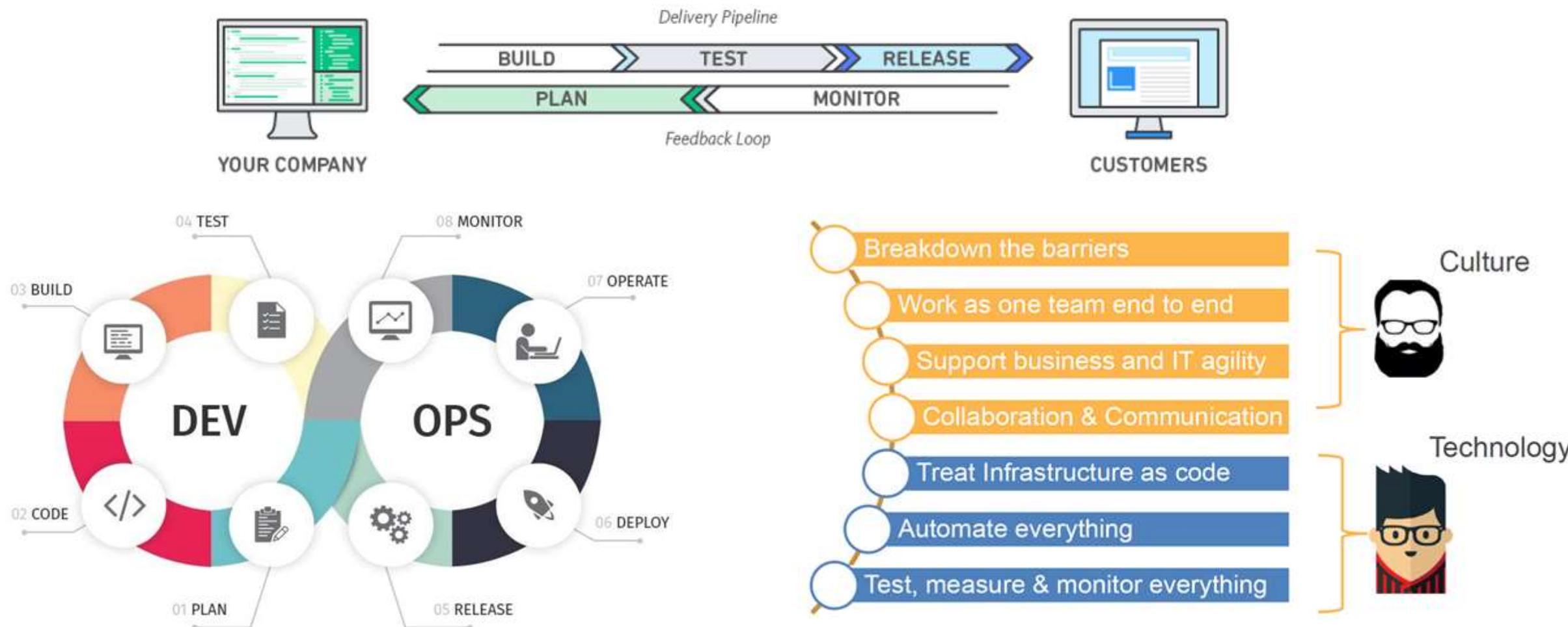
중앙 저장소(Repository)에서 코드와 이미지를 관리하고 플랫폼에 배포(Deployment)하는 형태로 지속적인 통합/배포(Continuous Integration/Deployment) 가능



Modified image from <https://medium.com/swlh/kubernetes-ci-cd-using-jenkins-on-google-cloud-5b10da6147a6>

# 하이브리드/멀티클라우드 운영 – DevOps

DevOps는 애플리케이션과 서비스를 빠른 속도로 제공할 수 있도록 조직의 역량을 향상시키는 문화 철학, 방식 및 도구의 조합

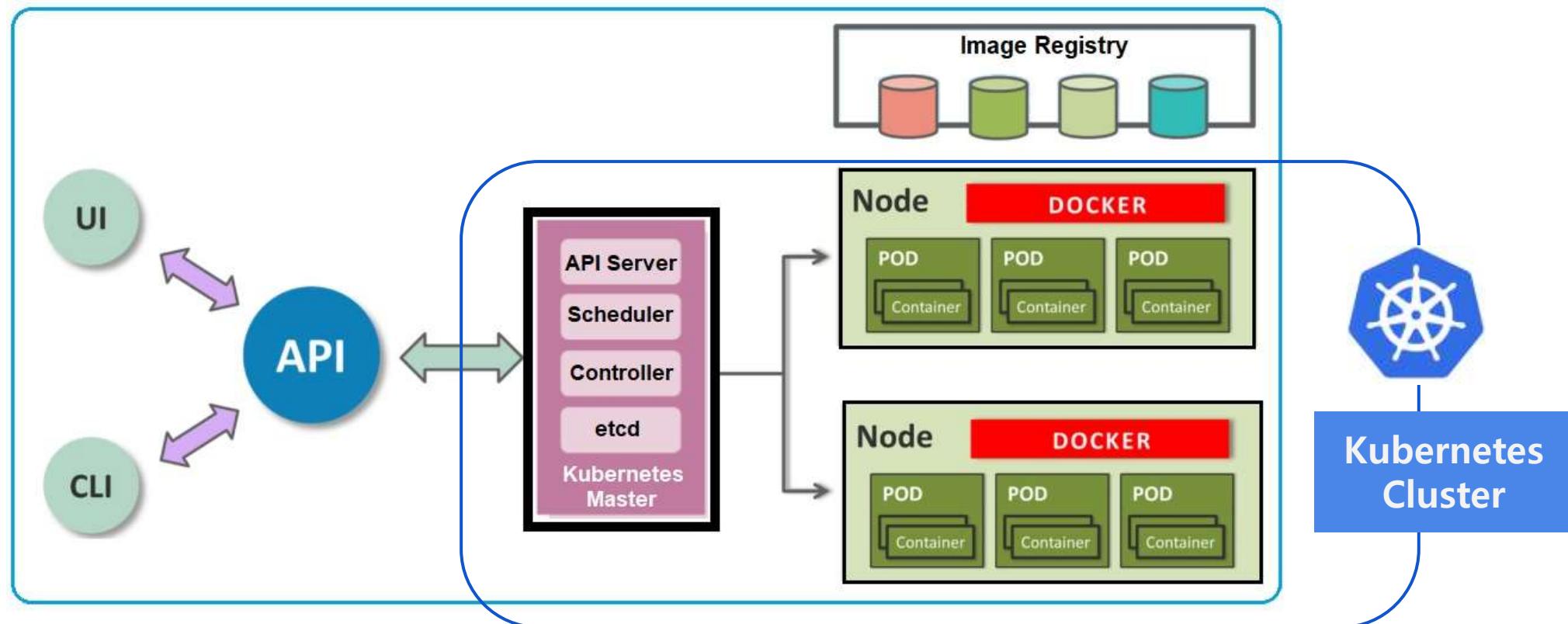


<https://aws.amazon.com/ko/devops/what-is-devops/>  
<https://dev.to/ashokisaac/devops-in-3-sentences-17c4>

# 쿠버네티스 (Kubernetes, k8s)

- 컨테이너 오케스트레이션(Orchestration) 기술 중 사실상 표준
- 완전관리형의 Kubernetes 서비스

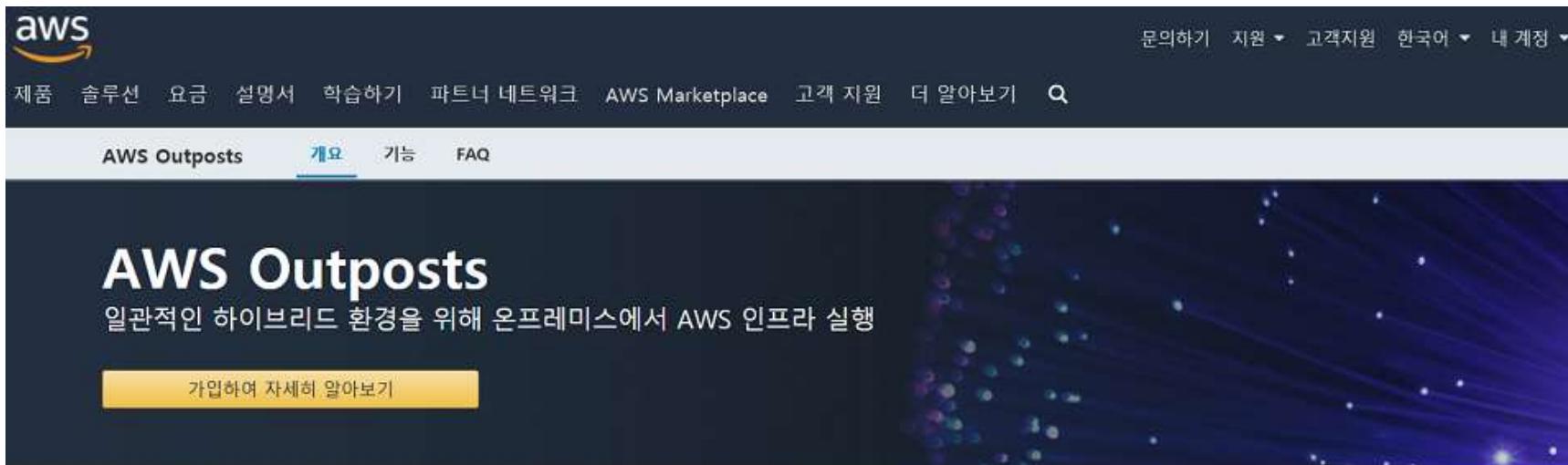
<https://aws.amazon.com/ko/eks/>



Reorganized from DZone (<https://dzone.com/articles/docker-containers-and-kubernetes-an-architectural>)

# 각 업체별 전략 - AWS

- **AWS Outposts : 1) VMWare Cloud on AWS Outposts , 2) AWS Outposts Native**



AWS Outposts는 네이티브 AWS 서비스, 인프라 및 운영 모델을 사실상 모든 데이터 센터, 코로케이션 공간 또는 온프레미스 시설로 옮길 수 있습니다. 온프레미스 및 클라우드에 걸쳐 동일한 API, 동일한 도구, 동일한 하드웨어 및 동일한 기능을 사용하여 일관된 하이브리드 환경을 제공할 수 있습니다. Outposts는 짧은 지연 시간 또는 로컬 데이터 처리 필요성에 따라 온프레미스에 유지되어야 하는 워크로드를 지원하는 데 사용할 수 있습니다.

AWS Outposts는 두 가지 변형으로 제공됩니다. 1) VMware Cloud on AWS Outposts를 통해 동일한 VMware 컨트롤 플레인 및 API를 사용하여 인프라를 실행할 수 있습니다. 2) AWS Outposts의 AWS 네이티브 변형을 통해 AWS 클라우드에서 실행하는 데 사용하는 것과 동일한 API 및 컨트롤 플레인을 온프레미스에서 사용할 수 있습니다.

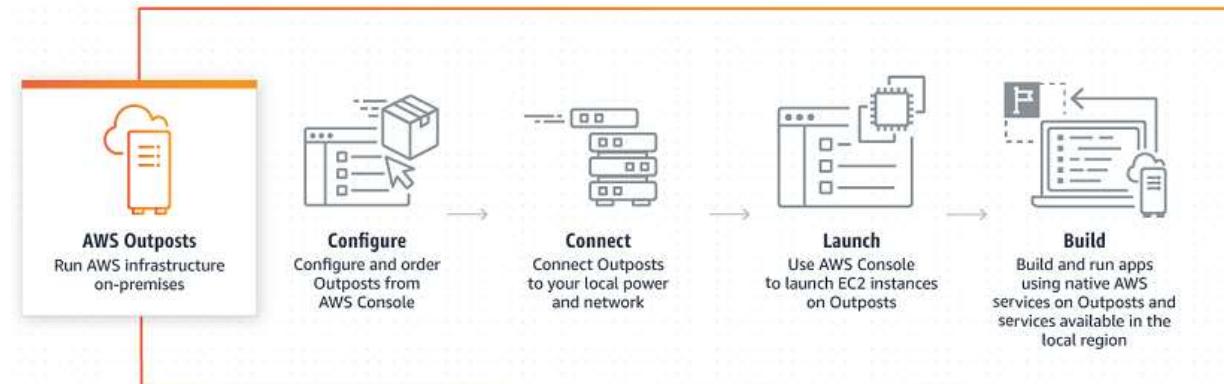
AWS Outposts 인프라는 AWS에서 유지 관리하고 지원하는 완전관리형이며, 최신 AWS 서비스에 대한 액세스를 제공합니다. 시작은 쉽습니다. AWS Management Console에 로그인하여 Outposts 서버를 주문하고 다양한 컴퓨팅 및 스토리지 옵션 중에서 선택할 수 있습니다. 하나 이상의 서버를 주문하거나 1/4, 절반 또는 전체 랙 유닛을 주문할 수 있습니다.



<https://aws.amazon.com/ko/outposts/>

# 각 업체별 전략 - AWS

- AWS Outposts : Using special hardware racks



<https://www.cloudmanagementinsider.com/aws-outposts-all-you-need-to-know/>  
<https://awsfeed.com/whats-new/aws-outposts-now-available-order-yours-today/>

# 각 업체별 전략 – MS Azure

- Azure Hybrid

The screenshot shows the Microsoft Azure homepage with a dark theme. At the top, there's a navigation bar with links for '개요', '솔루션', '제품', '설명서', '가격' (which is underlined in white), '교육', 'Marketplace', '파트너', '지원', '블로그', and '기타'. The main title 'Azure 하이브리드 혜택' is prominently displayed in large, bold, blue and white text. Below it, a sub-section title 'Azure에서 Software Assurance가 포함된 Windows Server 및 SQL Server 온-프레미스 라이선스 활용' is shown, also underlined in red. The main content area contains three bullet points: 1. Windows Server 및 SQL Server에 가장 적합한 클라우드인 Azure로 워크로드를 마이그레이션합니다. 2. AWS는 Windows Server 및 SQL Server용 Azure보다 5배 더 비쌉니다.\* 3. Azure Virtual Machines, Azure SQL Database PaaS 서비스 및 Azure Dedicated Host에 절감된 비용이 적용됩니다.

- ✓ Windows Server 및 SQL Server에 가장 적합한 클라우드인 Azure로 워크로드를 마이그레이션합니다.
- ✓ AWS는 Windows Server 및 SQL Server용 Azure보다 5배 더 비쌉니다.\*
- ✓ Azure Virtual Machines, Azure SQL Database PaaS 서비스 및 Azure Dedicated Host에 절감된 비용이 적용됩니다.

<https://azure.microsoft.com/ko-kr/pricing/hybrid-benefit/>

# 각 업체별 전략 – MS Azure

- Free Win-10 WSL(Windows Subsystem Linux) & Hyper-V

Microsoft | Docs Windows Azure Visual Studio Office Microsoft 365 .NET 자세히 ▾

Docs / Windows 10에 설치

제목으로 필터링

Windows 10에 Linux용 Windows 하위 시스템 설치 가이드

2018. 07. 23. • 읽는 데 2분 + 3

### Linux용 Windows 하위 시스템 설치

WSL용 Linux 배포판을 설치하려면 먼저 선택적인 "Linux용 Windows 하위 시스템" 기능을 사용하도록 설정해야 합니다.

- PowerShell을 관리자 권한으로 열어 실행합니다.
- 메시지가 표시되면 컴퓨터를 다시 시작합니다.

### 선택한 Linux 배포 설치



Microsoft | Store 오피스 Xbox 게임 Windows 게임 소프트웨어 학생 및 교직원 Xbox 기프트카드

Canonical Group Limited • 개발자 도구 > 서버

ubuntu

### Ubuntu 18.04 LTS

Ubuntu 18.04 on Windows allows one to use Ubuntu Terminal and run Ubuntu command line utilities including bash, ssh, git, apt and many more.

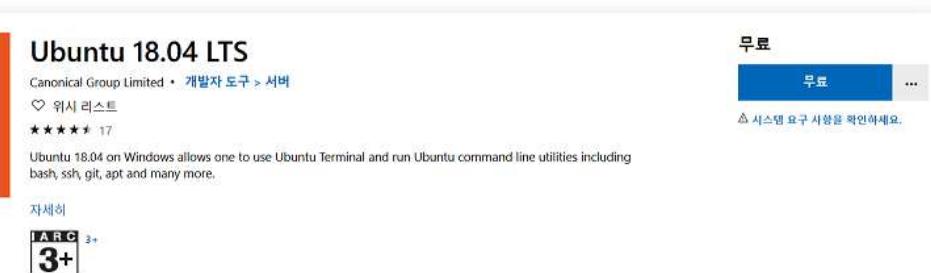
무료

무료 ...

△ 시스템 요구 사항을 확인하세요.

자세히

IARC 3+



Microsoft | Docs Windows Azure Visual Studio Office Microsoft 365 .NET 자세히 ▾

가상화 / Windows 10의 Hyper-V / Hyper-V 소개

제목으로 필터링

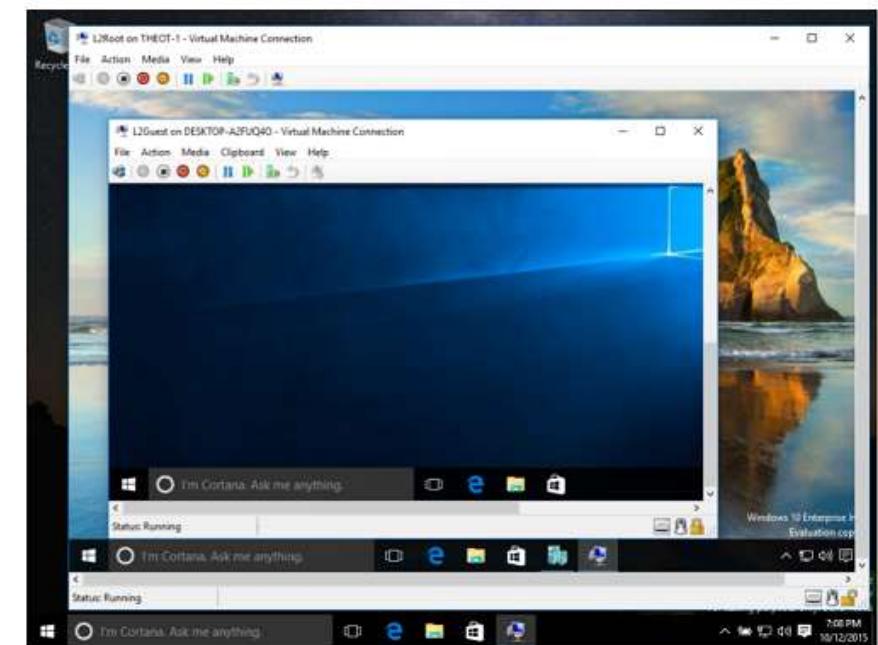
### Hyper-V 소개

- Hyper-V 설치
- Virtual Machine 만들기
- > Hyper-V가 설치된 가장 컴퓨터 관리
- > Hyper-V 호스트 관리
- > 참고자료
- > 커뮤니티 및 지원

## Windows 10의 Hyper-V 소개

2018. 06. 25. • 읽는 데 5분 + 3

많은 소프트웨어 개발자, IT 전문가 또는 기술 매니아들은 여러 운영 체제를 실행해야 합니다. Hyper-V를 사용하면 Windows에서 가장 마신으로 여러 운영 체제를 실행할 수 있습니다.



<https://docs.microsoft.com/ko-kr/windows/wsl/install-win10>  
<https://www.microsoft.com/ko-kr/p/ubuntu-1804-lts/9n9tngvndl3q?activetab=pivot:overviewtab>  
<https://docs.microsoft.com/ko-kr/virtualization/hyper-v-on-windows/about/>

# 각 업체별 전략 – Google GCP

- 멀티클라우드 모니터링을 완벽 지원. 3<sup>rd</sup> Party Solution을 제외하고는 사실상 유일.

## Stackdriver

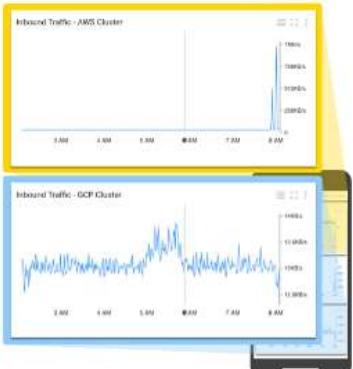
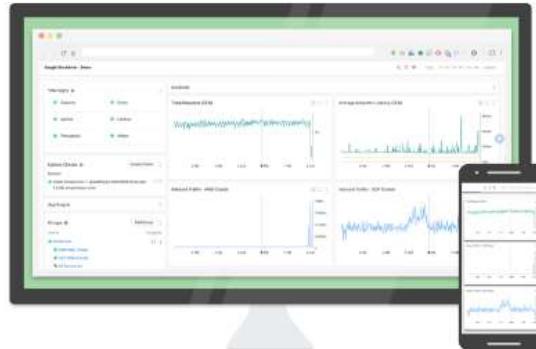
서비스, 컨테이너, 애플리케이션, 인프라를 모니터링 및 관리합니다.

무료로 사용해 보기

이 제품의 문서 보기

### 코드 및 애플리케이션을 완벽하게 관찰

Stackdriver는 측정항목, 로그, 이벤트를 인프라에서 집계하고 개발자와 운영자에게 관측 가능한 신호를 다양하게 제공하여 근본 원인 분석의 속도를 높이고 평균 문제 해결 시간(MTTR)을 단축합니다. Stackdriver는 광범위한 통합 또는 여러 개의 '관리 창'이 필요하지 않으며 개발자가 특정 클라우드 제공업체만 사용하도록 강제하지 않습니다.

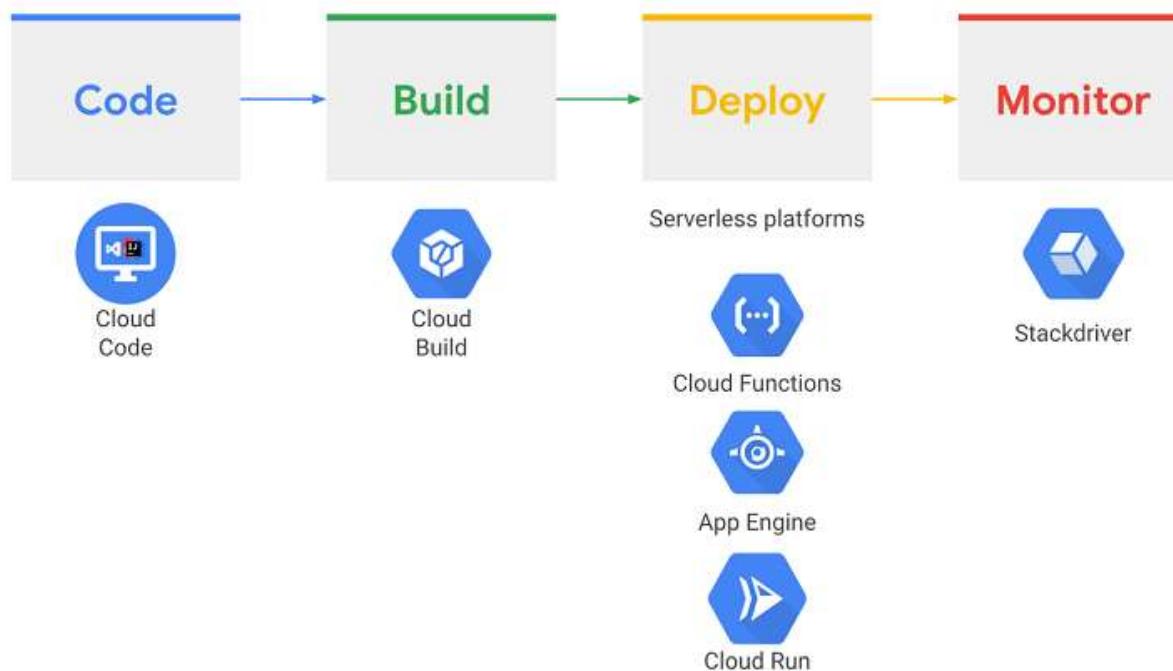


### 여러 클라우드 및 온프레미스 인프라에서 작동

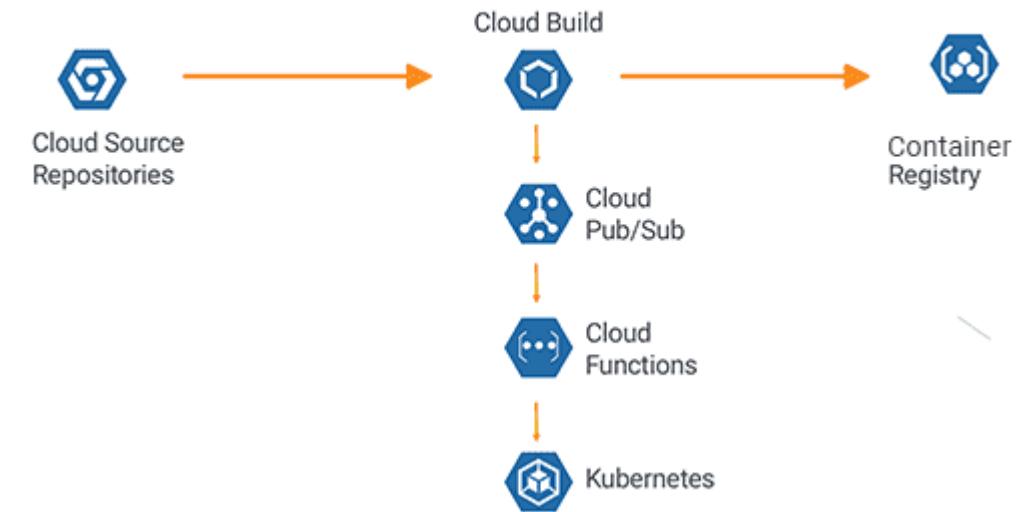
Stackdriver는 처음부터 클라우드 기반 애플리케이션을 위해 구축되었습니다. Google Cloud Platform, Amazon Web Services, 온프레미스 인프라, 하이브리드 클라우드 등, 실행 환경에 관계없이 모든 클라우드 계정 및 프로젝트의 측정항목, 로그, 메타데이터를 모아 포괄적인 단일 뷰에서 환경을 확인할 수 있으므로 서비스 동작을 빠르게 이해하고 조치를 취할 수 있습니다.

# 각 업체별 전략 – Google GCP

- CI/CD DevOps Pipeline



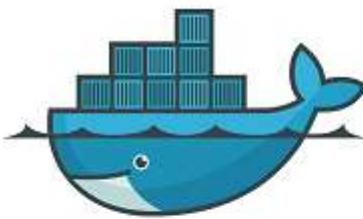
## DevOps Pipeline on Google Cloud Platform



# 각 업체별 전략 – Google GCP Cloud Run

- 컨테이너를 프로덕션으로 신속히 배포 (Stateless, Serverless)
- 어디서든 동일한 환경 이용 가능 (On-Prem, GCP, Anthos)
- 100% 오픈소스 기반

Docker  
Container



Kubernetes



Cloud Run



Knative



Knative



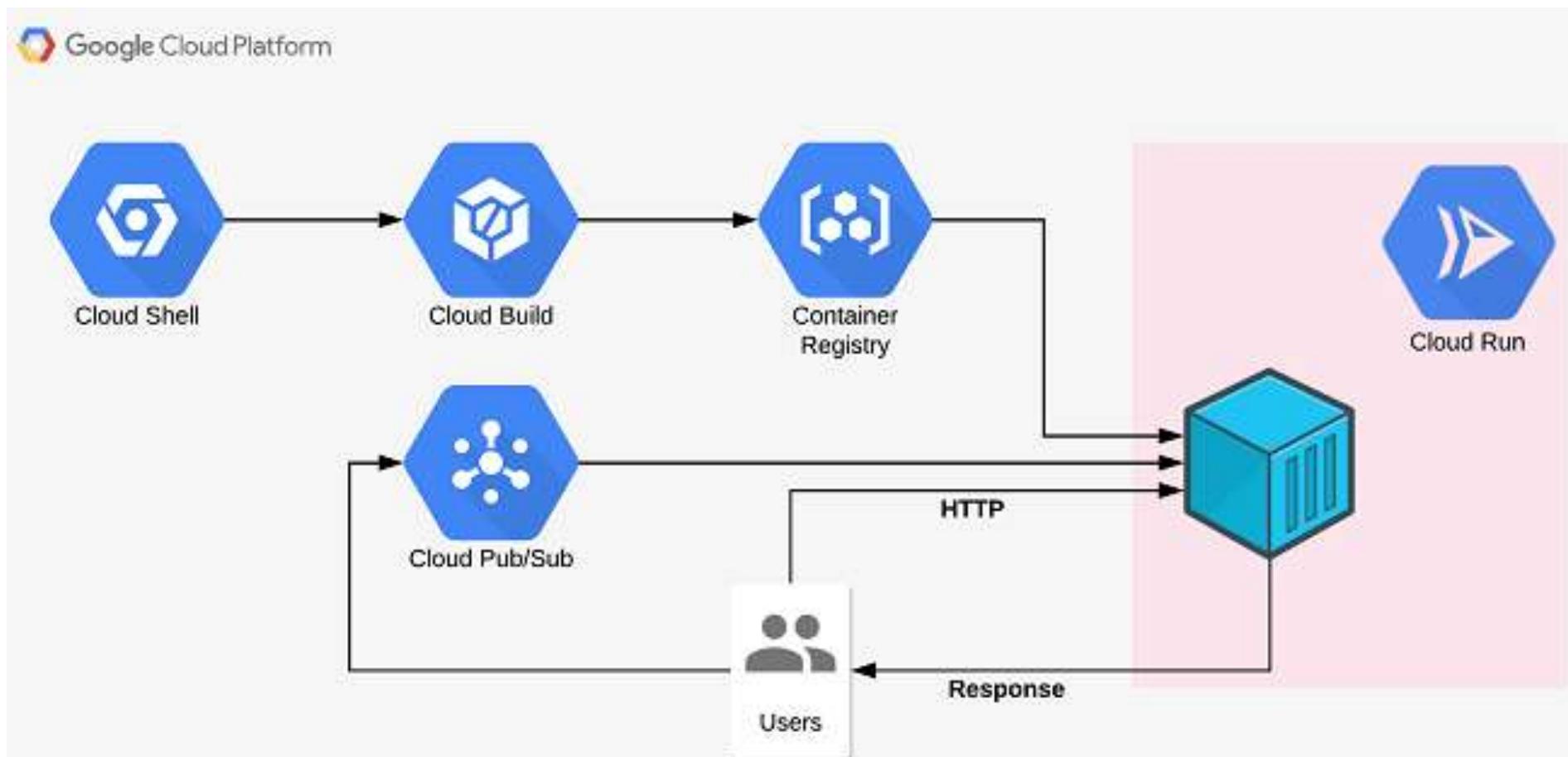
GKE  
(GCP Kubernetes  
Engine)

<https://cloud.google.com/run/?hl=ko>

<https://knative.dev/>

<https://medium.com/google-cloud/knative-to-cloud-run-f0ed1617e256>

# 각 업체별 전략 – Google GCP Cloud Run

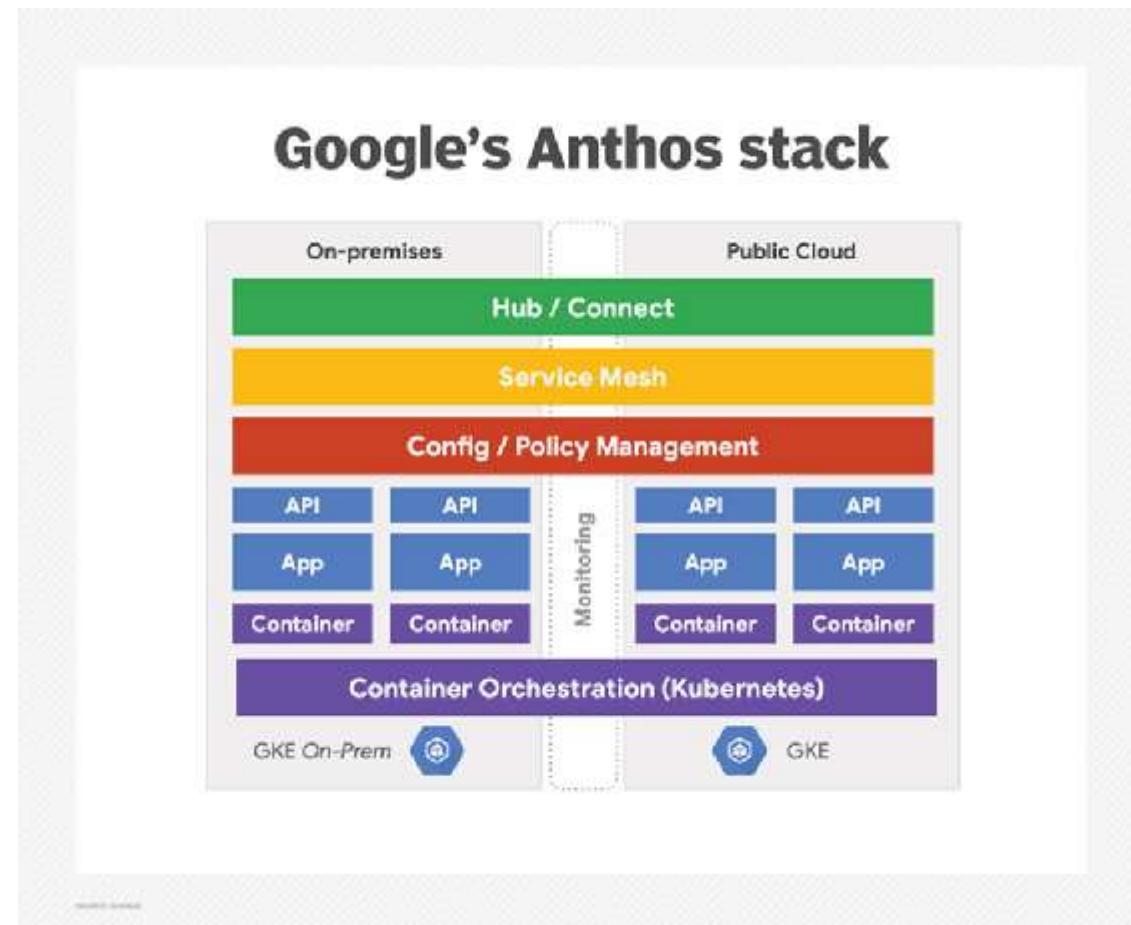
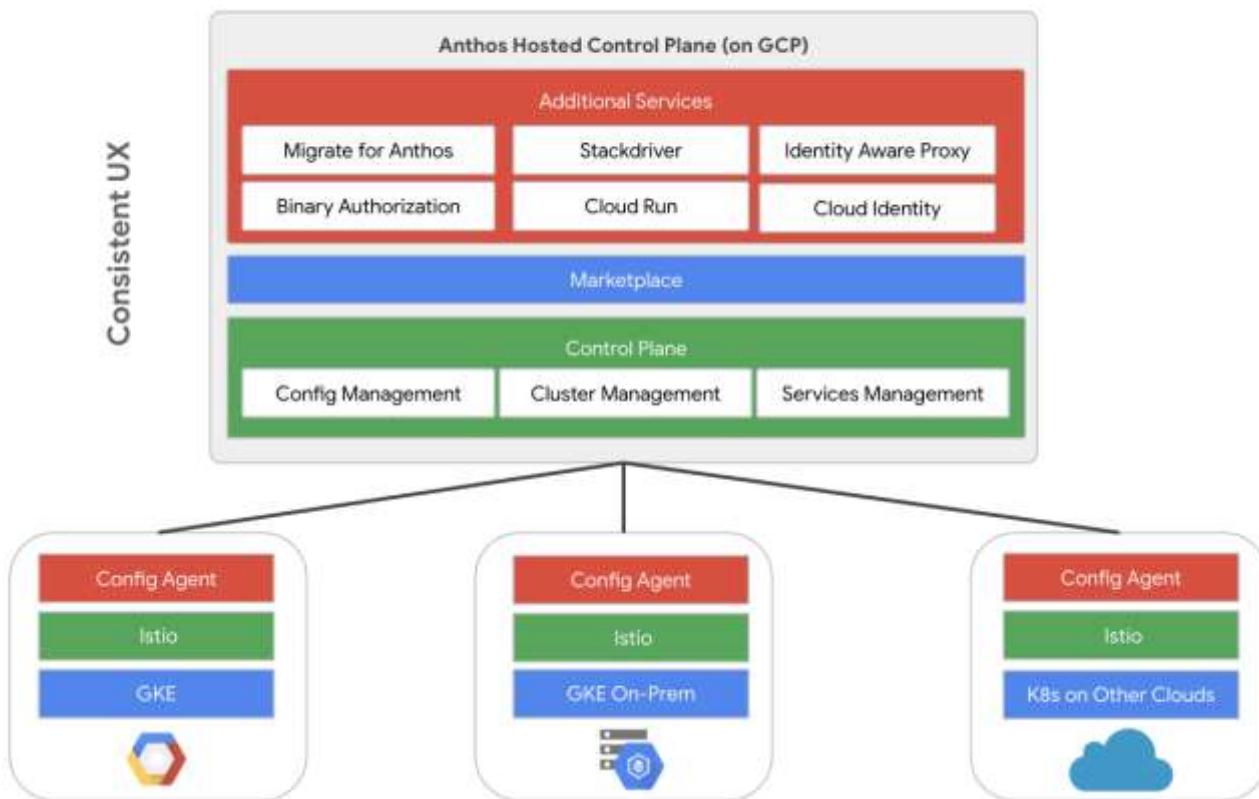


<https://towardsdatascience.com/cloud-run-dataset-summaries-via-http-request-9b5fe24fe9c1>

# 각 업체별 전략 – Google GCP Anthos



## Anthos: Hybrid Ecosystem



<https://codelabs.developers.google.com/codelabs/anthos-workshop/#0>

# 각 업체별 전략 – Google Colaboratory

Colaboratory에 오신 것을 환영합니다

파일 수정 보기 삽입 런타임 도구 도움말

공유 ▾ 설정 가능 ▾

목차 X + 코드 + 텍스트 드라이브로 복사

연결 ▾ 수정 가능 ▾

시작하기

데이터 과학  
머신러닝  
추가 리소스  
머신러닝 예제  
섹션

Colaboratory란?

Colaboratory(또는 줄여서 'Colab')를 사용하면 브라우저에서 Python을 작성하고 실행할 수 있습니다.

- 구성 필요 없음
- GPU 무료 액세스
- 간편한 공유

학생이든, 데이터 과학자든, AI 연구원이든 Colab으로 업무를 더욱 간편하게 처리할 수 있습니다. [Colab 소개 영상](#)에서 자세한 내용을 확인하거나 아래에서 시작해 보세요.

시작하기

지금 읽고 계신 문서는 정적 웹페이지가 아니라 코드를 작성하고 실행할 수 있는 대화형 환경인 **Colab 메모장**입니다.

예를 들어 다음은 값을 계산하여 변수로 저장하고 결과를 출력하는 간단한 Python 스크립트가 포함된 **코드 셀**입니다.

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

86400

<https://colab.research.google.com>

# 각 업체별 전략 – Google CodeLabs

The screenshot shows the homepage of the Google Codelabs website. At the top, there is a dark header bar with the "Codelabs" logo and a search bar. Below the header, a large "Welcome to Codelabs!" heading is displayed. A descriptive paragraph explains what Codelabs are: "Google Developers Codelabs provide a guided, tutorial, hands-on coding experience. Most codelabs will step you through the process of building a small application, or adding a new feature to an existing application. They cover a wide range of topics such as Android Wear, Google Compute Engine, Project Tango, and Google APIs on iOS." Below this text is a link to "CodeLab tools on GitHub".

The main content area displays a grid of six Codelab projects:

Project Title	Duration	Last Updated	Start Button
10 Tips to make Ad Monetization Smarter with Firebase	79 min	Updated Feb 28, 2019	Start
A Tour of Cloud IoT Core		Updated Dec 7, 2018	Start
AI on a microcontroller with TensorFlow Lite and SparkFun Edge	28 min	Updated Oct 23, 2019	Start
ARCore Augmented Faces (Android)		Updated Jan 10, 2020	Start
ARCore Augmented Images	34 min	Updated Sep 11, 2019	Start
ARCore Cloud Anchors		Updated Sep 11, 2019	Start

<https://codelabs.developers.google.com>

## Module 10

# Multi-Cloud Architecture 설계 원리 및 사례

# **Part I**

# **AWS Well-Architected Framework**

# Well-Architected Framework



운영 우수성



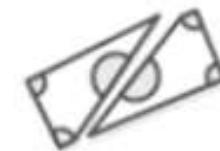
보안



신뢰성

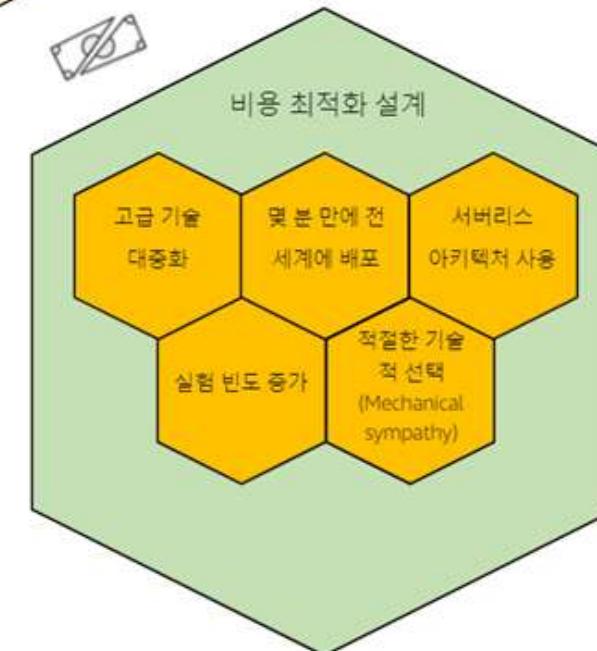
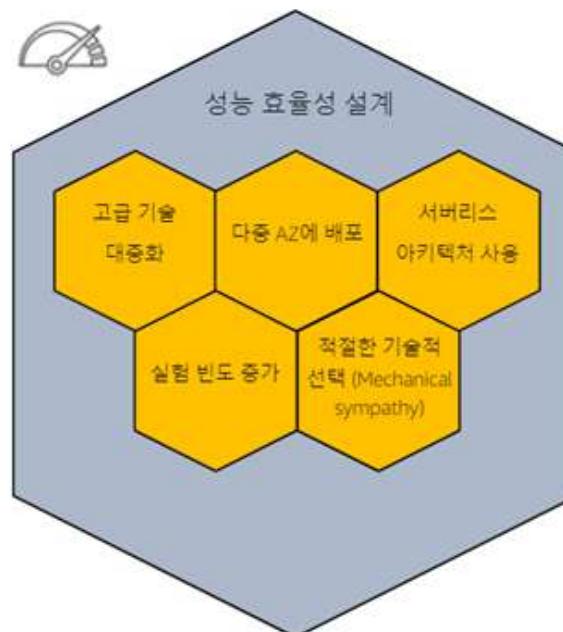
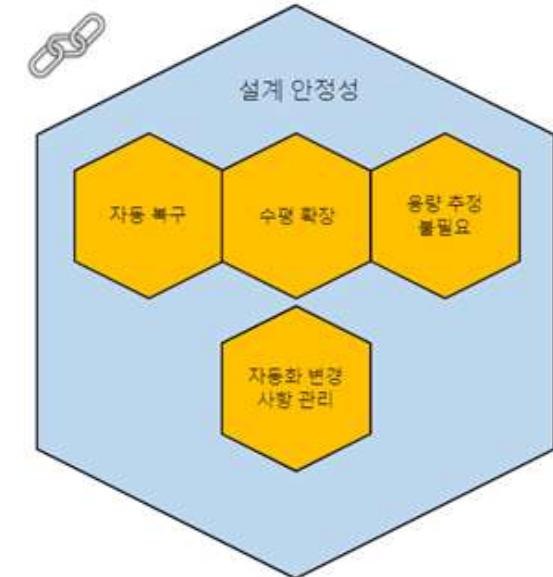
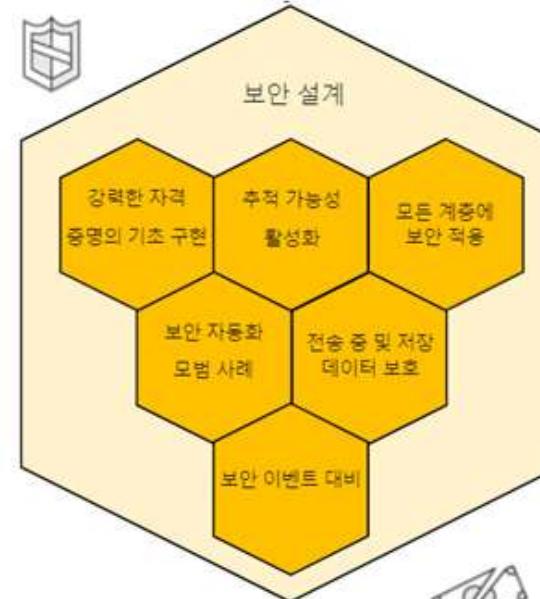
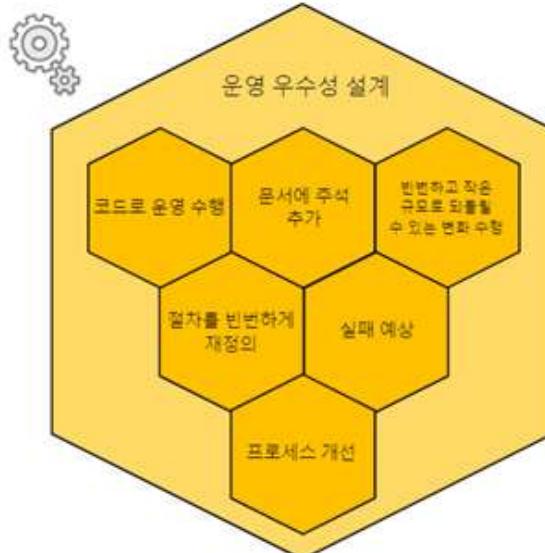


성능 효율성



비용 최적화

# Well-Architected Framework



# Well-Architected Framework

## Well-Architected Framework

- ❑ Developed through reviewing customers' architectures on AWS.
- ❑ Helps build the most...
  - ✓ Secure
  - ✓ High-performing
  - ✓ Resilient
  - ✓ Efficient
- ❑ Consistent approach for evaluating architectures
- ❑ Five pillars with design principles

## Well-Architected Framework

 Build and deploy faster

 Lower or mitigate risks

 Make informed decisions

 Learn AWS best practices

[aws.amazon.com/architecture/well-architected/](https://aws.amazon.com/architecture/well-architected/)

## The Five Pillars



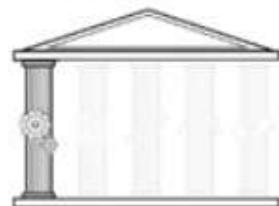
- ❑ Operational Excellence
- ❑ Security
- ❑ Reliability
- ❑ Performance Efficiency
- ❑ Cost Optimization

<https://aws.amazon.com/ko/architecture/well-architected/>

<https://aws.amazon.com/ko/blogs/korea/aws-well-architected-framework-in-korean/>

# 첫번째 기둥(1<sup>st</sup> Pillar) : 운영 우수성 (Operational Excellence)

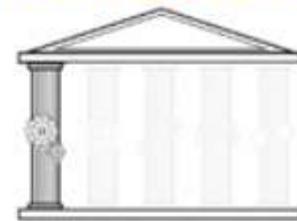
## Operational Excellence



### Overview

- ❑ Run and monitor systems to deliver business value
- ❑ Continually improve supporting processes and procedures

## Operational Excellence



### Design Principles

1. Perform operations as code
2. Annotate documentation
3. Make frequent, small, reversible changes
4. Refine operations procedures frequently
5. Anticipate failure
6. Learn from all operational failures

# 두번째 기둥(2<sup>nd</sup> Pillar) : 보안 (Security)



- ❑ Protect:
  - ✓ Information
  - ✓ Systems
  - ✓ Assets
- ❑ Risk assessments and mitigation strategies



## Design Principles

1. Implement a strong identity foundation
2. Enable traceability
3. Apply security at all layers
4. Automate security best practices
5. Protect data in transit and at rest
6. Prepare for security events

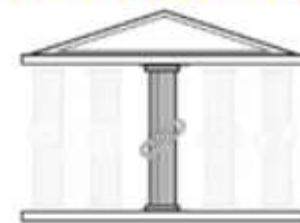
# 세번째 기둥(3<sup>rd</sup> Pillar) : 안정성 (Reliability)



## Overview

- ❑ Recover from infrastructure or service disruptions
- ❑ Dynamically acquire computing resources to meet demand
- ❑ Mitigate disruptions

## Reliability

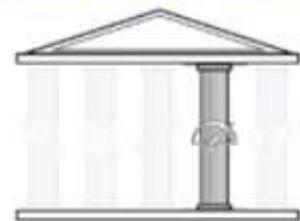


## Design Principles

1. Test recovery procedures
2. Automatically recover from failure
3. Scale horizontally to increase aggregate system availability
4. Stop guessing capacity
5. Manage change in automation

# 네번째 기둥(4<sup>th</sup> Pillar) : 성능 효율성 (Performance Efficiency)

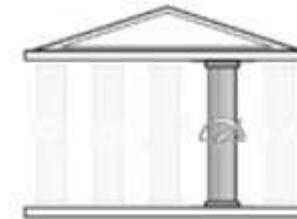
## Performance Efficiency



### Overview

- ❑ Use computing resources efficiently
- ❑ Maintain efficiency as demand changes and technologies evolve

## Performance Efficiency

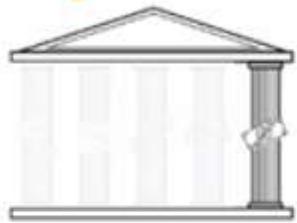


### Design Principles

1. Democratize advanced technologies
2. Go global in minutes
3. Use serverless architectures
4. Experiment more often
5. Mechanical sympathy

# 다섯번째 기둥(5<sup>th</sup> Pillar) : 비용 최적화 (Cost Optimization)

## Cost Optimization



### Overview

- Avoid or eliminate unneeded cost or suboptimal resources

## Cost Optimization



### Design Principles

1. Adopt a consumption model
2. Measure overall efficiency
3. Stop spending money on data center operations
4. Analyze and attribute expenditure
5. Use managed services to reduce cost of ownership

# **내결함성 및 고가용성 설계**

## **(Fault-Tolerant and High-Availability Architecture)**

# Fault Tolerance and High Availability

## 내결함성

- 시스템이 작동 가능 상태를 유지할 수 있는 능력
- 애플리케이션 구성요소의 내장된 중복성

## Fault Tolerance



- Remain operational even if components fail
- Built-in redundancy of an application's components

## 고가용성

- 시스템이 작동하고 액세스 가능한 상태 유지
- 가동 중단을 최소화
- 필요한 인적 개입을 최소화

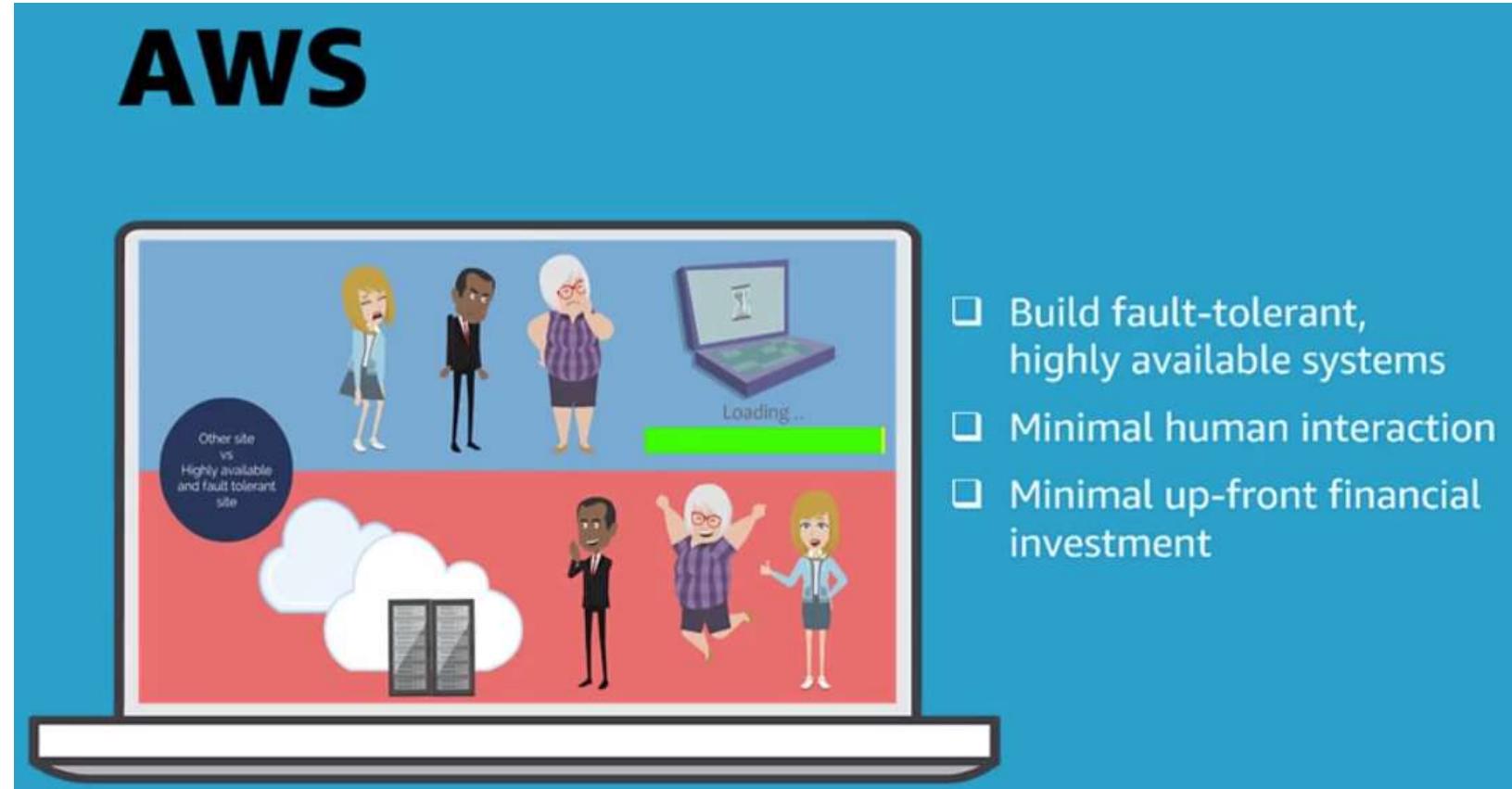
## High Availability



- “Always” functioning and accessible
- Downtime is minimized
- Without human intervention

# On Premises vs AWS

- 기존(온프레미스)
  - 높은 비용
  - 미션 크리티컬 애플리케이션만 해당
- AWS
  - 여러 대의 서버
  - 가용 영역
  - 리전
  - 내결함성 서비스



# 고가용성 서비스와 도구

- ✓ Amazon S3 및 Amazon Glacier
- ✓ DynamoDB
- ✓ Amazon CloudFront
- ✓ Amazon SWF
- ✓ Amazon SQS
- ✓ Amazon SNS
- ✓ Amazon SES
- ✓ Amazon Route 53
- ✓ Elastic Load Balancing
- ✓ IAM
- ✓ Amazon CloudWatch

- ✓ Amazon CloudSearch
- ✓ AWS Data Pipeline
- ✓ Amazon Kinesis
- ✓ Auto Scaling
- ✓ Amazon Elastic File System
- ✓ AWS CloudFormation
- ✓ Amazon WorkMail
- ✓ AWS Directory Service
- ✓ AWS Lambda
- ✓ Amazon EBS
- ✓ Amazon RDS

기본적으로 HA 지원 서비스

- Amazon EC2
- Amazon VPC
- Amazon Redshift
- Amazon ElastiCache
- AWS Direct Connect

\* 여기에 나열된 서비스가 전부는 아닙니다.

적절한 아키텍처를 통해 HA 지원



탄력적 로드 밸런서



탄력적 IP 주소



Amazon Route 53



Auto Scaling



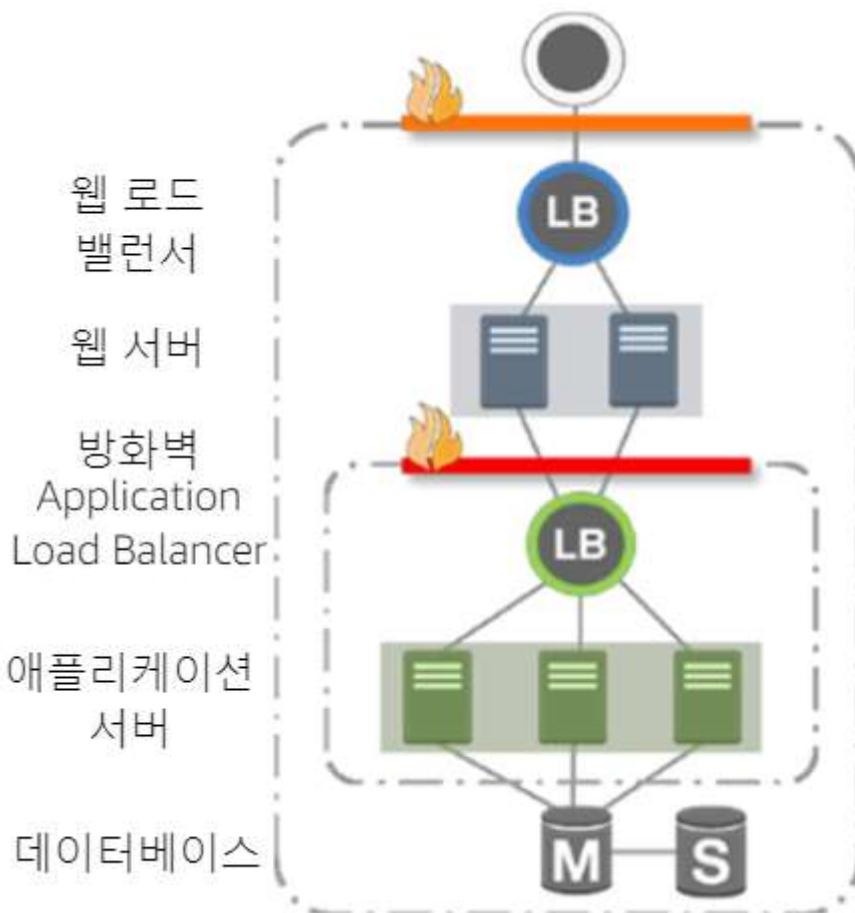
Amazon CloudWatch

# 내결함성 및 고가용성 구성

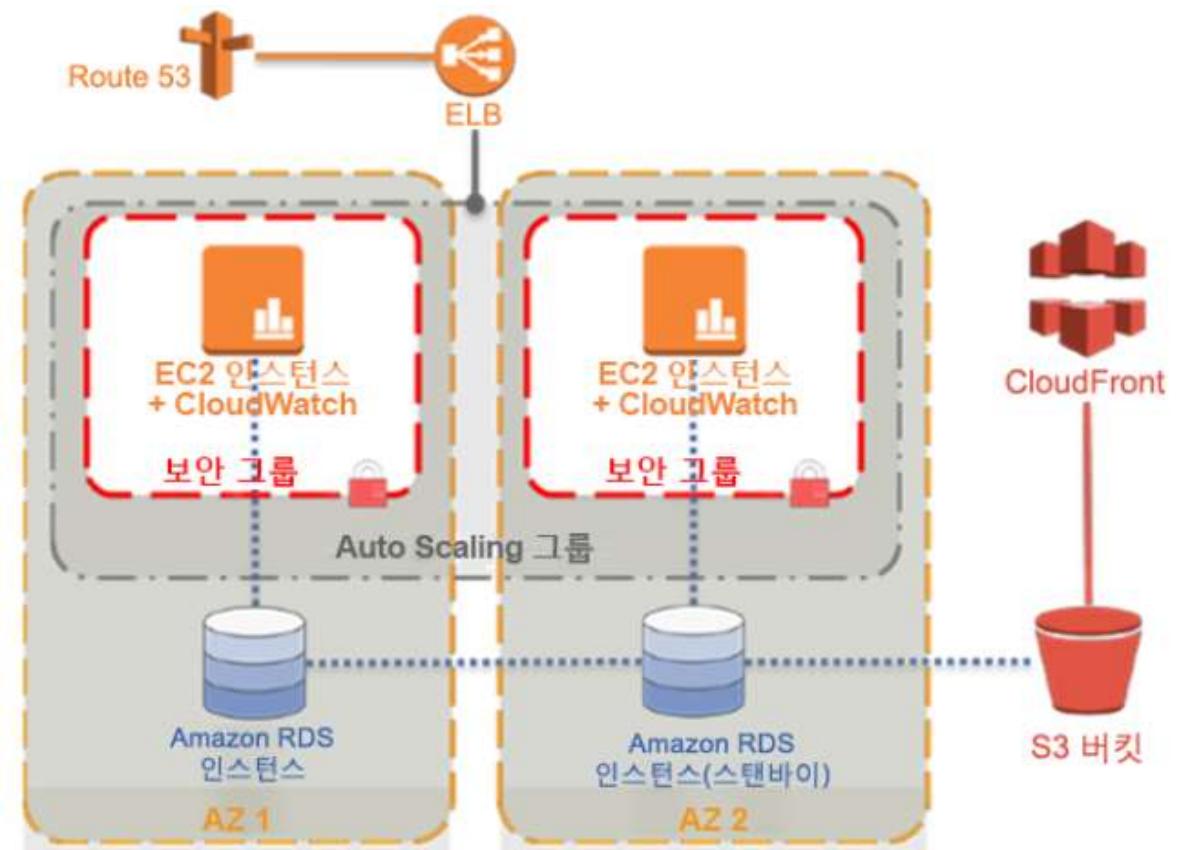


# On Premises vs AWS 아키텍처

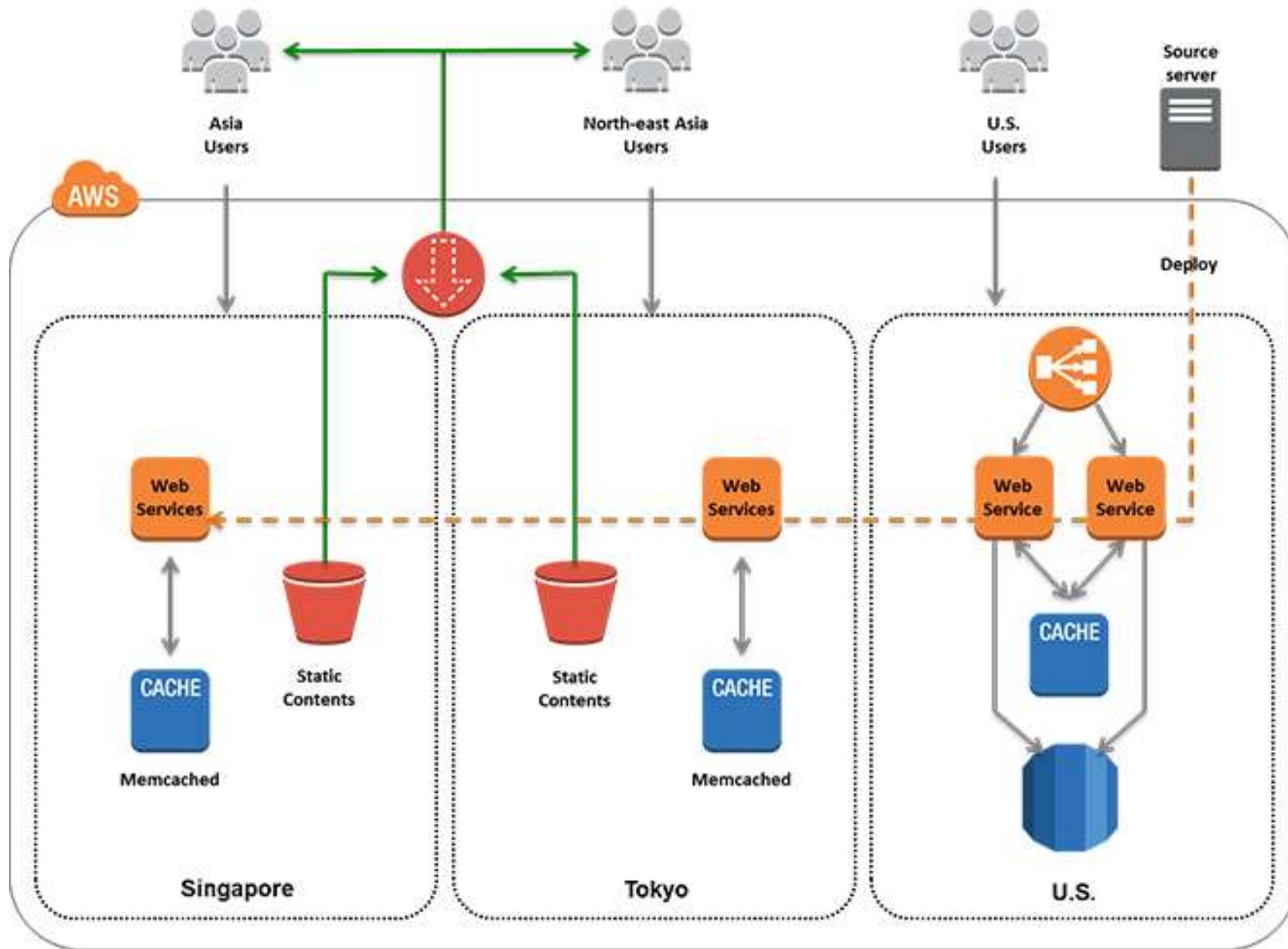
기존 아키텍처



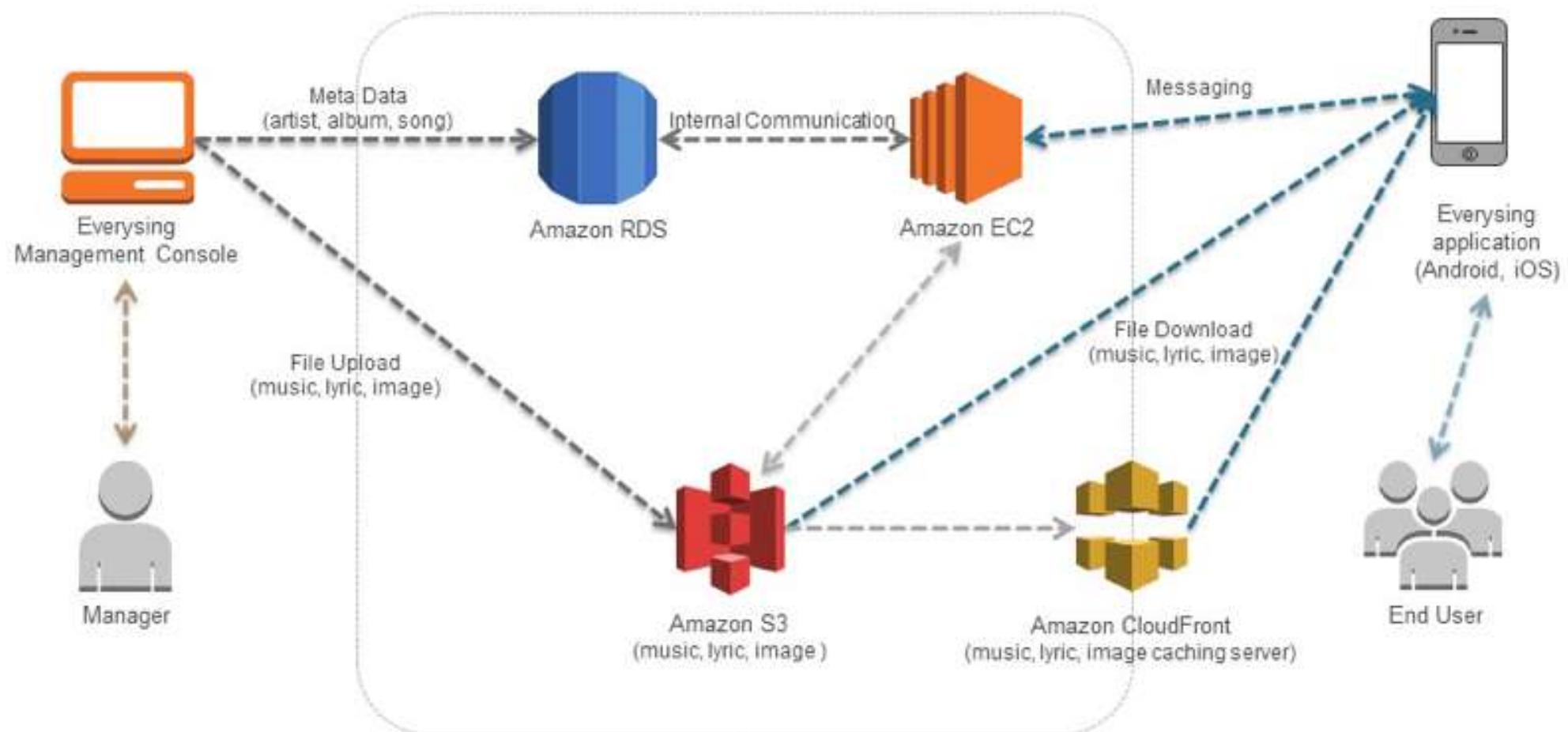
AWS 아키텍처



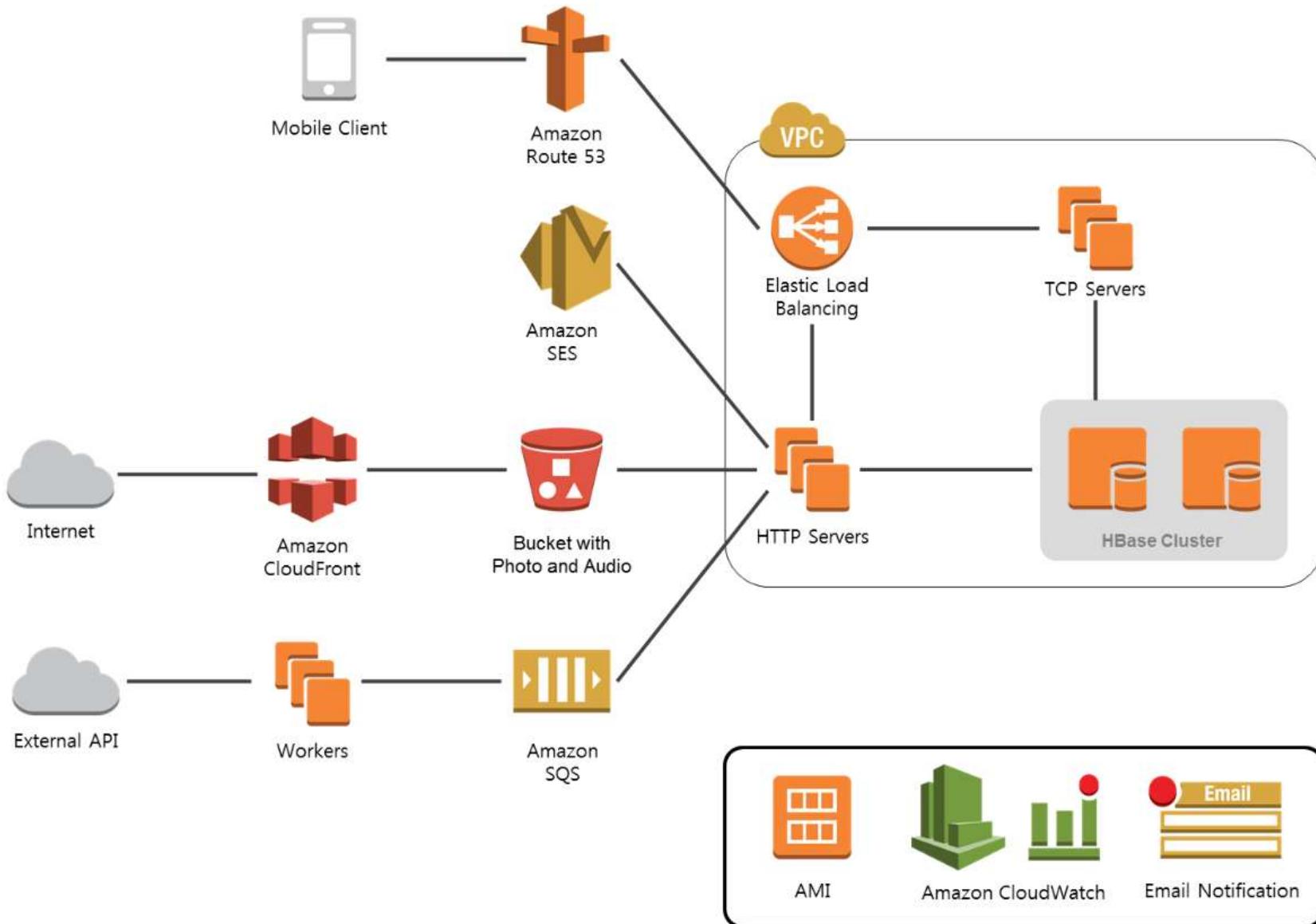
# A사 사례



# S사 사례

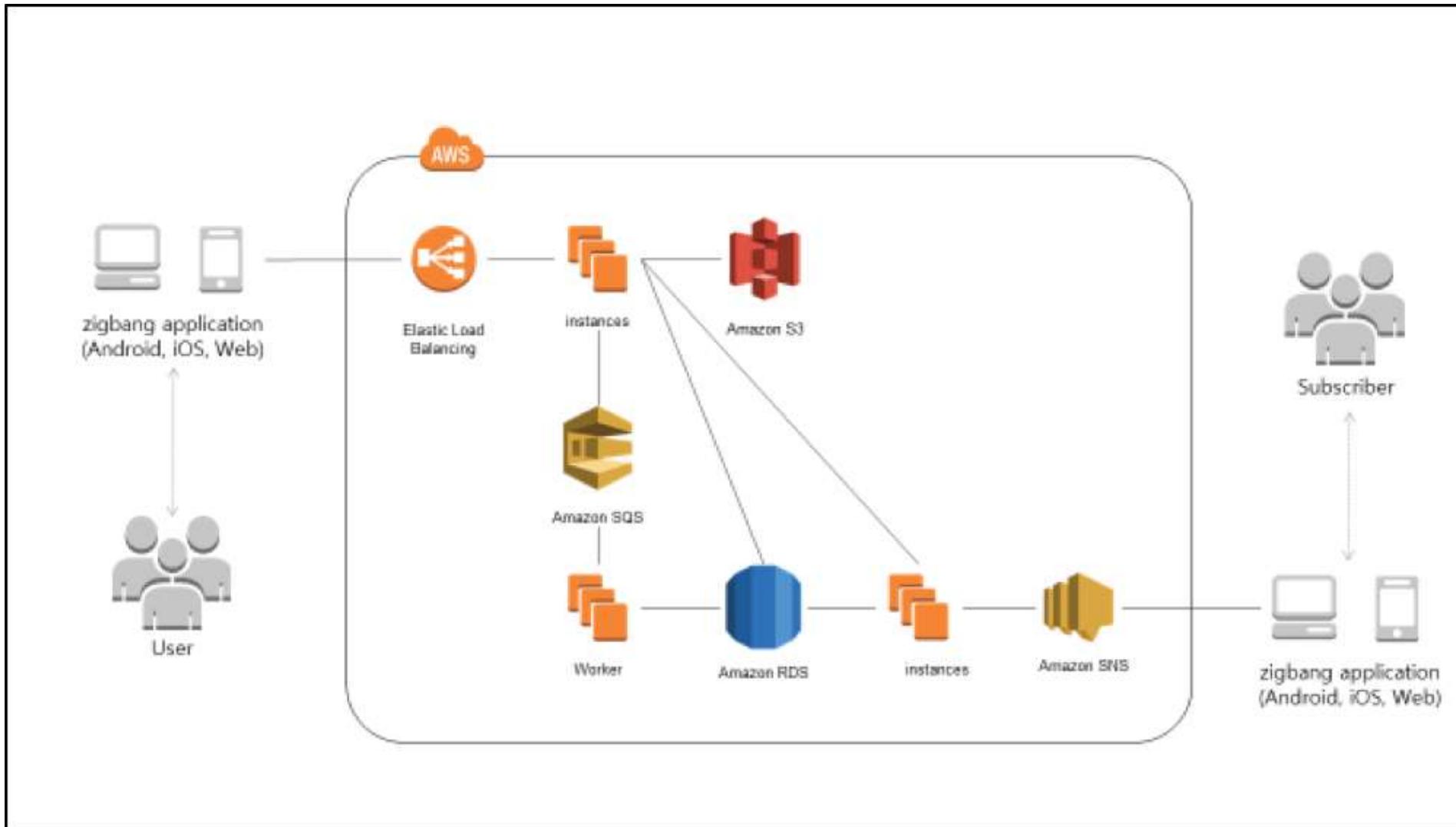


# V사 사례



<https://aws.amazon.com/ko/solutions/case-studies/vcnc/>  
<https://between.us/>

# Z사 사례



<https://aws.amazon.com/ko/solutions/case-studies/zigbang/>



## Q & A

# Thank You !

