



# “오픈소스”로 만들어가는 “멀티클라우드” 생태계

## 클라우드바리스타 커뮤니티 제6차 컨퍼런스

[세션] CB-Larva

### 멀티클라우드 인프라 및 응용을 위한 네트워크

김 윤 곤

CB-Larva 인큐베이터 리더

카페라떼(Cafe Latte) 한잔 어떠세요 ?

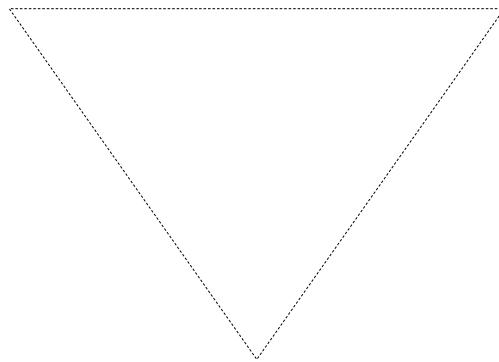


I'm serious about

# Cost

# Price

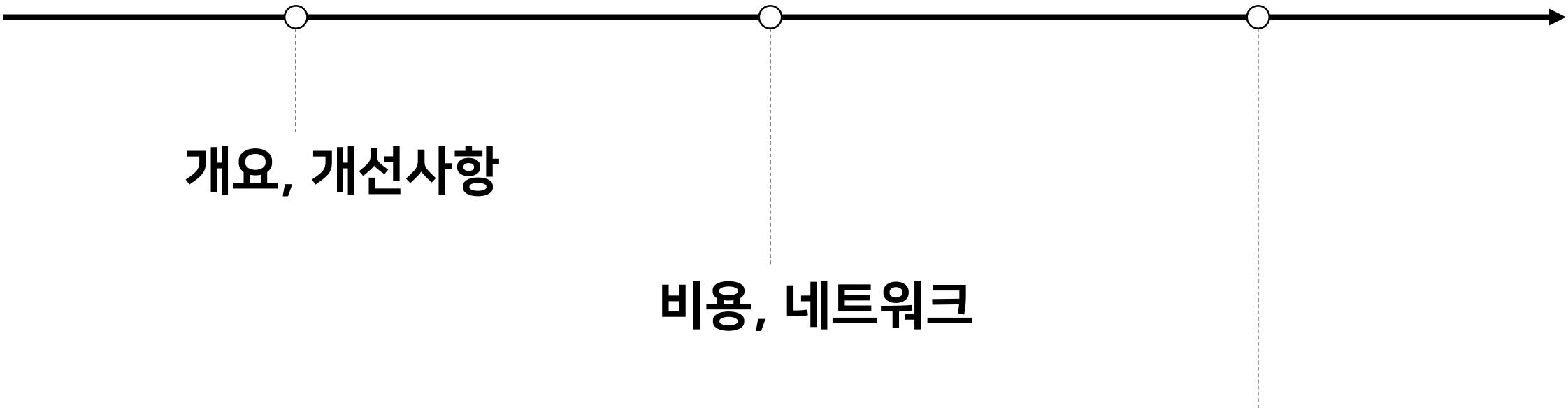
It's important but ...



Multi-Cloud Network

# 이번 세션은?

멀티클라우드 인프라를 위한 가상 네트워크 기술:



**기술 소개**  
(차기 컨퍼런스의 예고편 ^^;;)



# 이번 세션은?

응용/도메인/기관 특화 SW



CLOUD  
BARISTA

멀티클라우드 서비스 개방형 인터페이스

멀티클라우드 애플리케이션 실행환경  
통합관리 프레임워크

멀티클라우드 인프라 서비스  
통합 관리 프레임워크

멀티클라우드 인프라 연동  
프레임워크

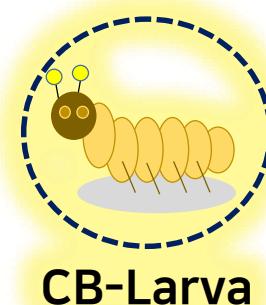
멀티클라우드  
서비스  
통합  
관리  
프레임워크

멀티클라우드 서비스 공통 플랫폼

CB-Larva: Cloud-Barista Incubator



멀티클라우드 기술 실험실



CB-Larva

Cloud-Barista의 정체성을 잃지 않고,  
지속적으로 신규 니즈를 수용하기 위하여  
신기술, 부족기술 등의 POC를 수행하며,  
이를 Cloud-Barista로 흡수하기 위한 기술 인큐베이터

# 목 차

---

I

관점을 바꿔서, 멀티클라우드 네트워크 개요

II

한번쯤 볼만한, 멀티클라우드 데이터 전송 과금(Data transfer billing)

III

멀티클라우드 인프라 및 응용을 위한, 네트워크 기술 소개

IV

함께 만들어가는, 공개 SW

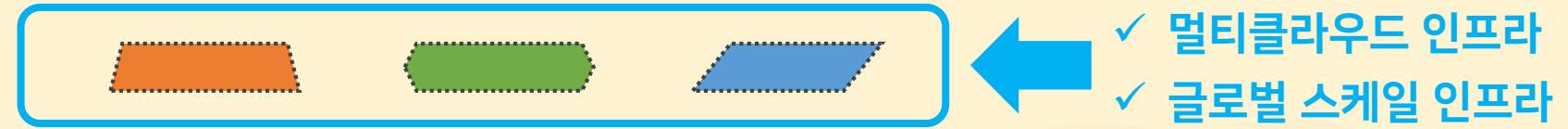
V

부록, 시스템 구조 및 주요 기술 설명

# 관점을 바꿔서, 하나의 인프라

## 우리의 서비스 / 우리의 플랫폼

#관점을-바꿔서 #하나의-인프라



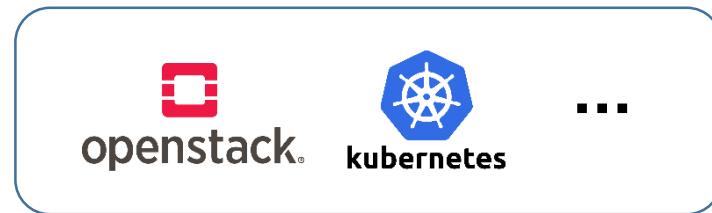
# 도전과제(Challenge)

#시작점 #멀티클라우드 #가상 #네트워크

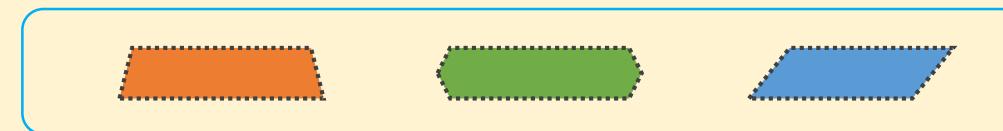
글로벌 스케일 인프라에

OpenStack, Kubernetes 등 기존 도구를 설치하고 통합 운용하고 싶습니다.

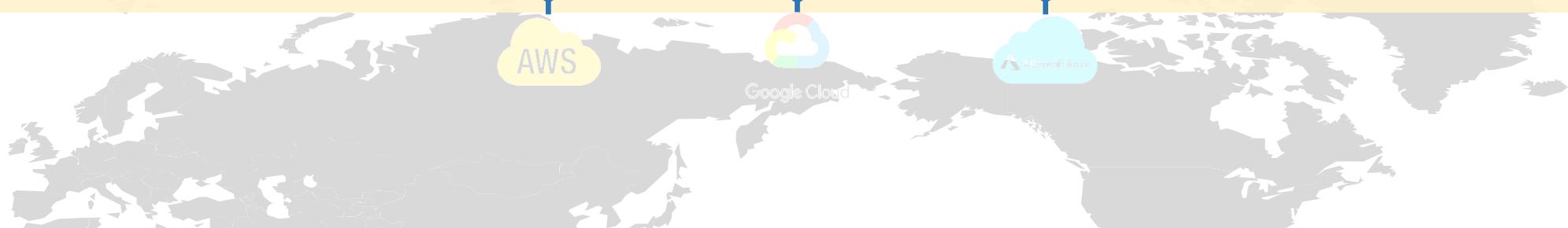
기본적으로 사설 네트워크/동일 서브넷 상에 설치를 요구하는 도구/응용은 글로벌 스케일 인프라에 설치하기 어려움



튜닝하여 설치  
또는  
동일 서브넷 구성하여 설치



- ✓ 멀티클라우드 인프라
- ✓ 글로벌 스케일 인프라



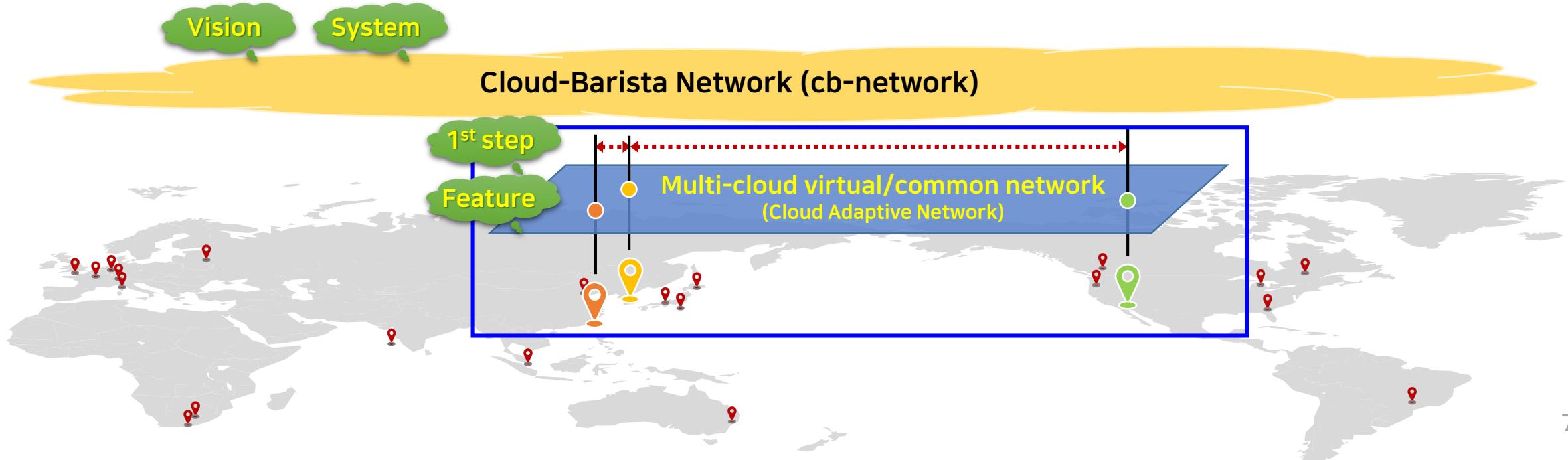
# 글로벌 스케일 네트워킹

(멀티클라우드 인프라 및 응용의 통신을 위하여! ^^)

전세계 클라우드 인프라를 엮기 위해 여러 클라우드 네트워크의 “다양성”과 “가변성”을 해결하는 효율적인 전송 방법



- ✓ 클라우드 서비스 사업자(CSP), 지역(Region), 구역(Zone)에 따라 “클라우드 네트워크” 구성이 상이함
- ✓ 정책 및 상황에 따라 가상머신(VM)/컨테이너에 할당되는 네트워크 정보가 변경 될 수 있음(예, IP 변경)
- ✓ 멀티클라우드 상의 가상머신(VM)/컨테이너 간 거리 및 통신 속도가 매우 가변적임

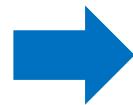


# Cloud Adaptive Network ?

Cloud Adaptive Network (CLADNet): 멀티클라우드 가상/공통 네트워크 기술

clad: 차려입은, 장비한

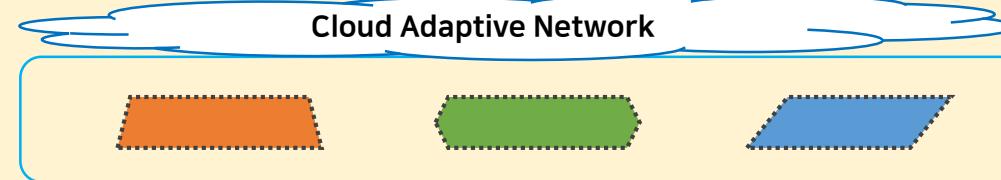
동일 서브넷 구성하여 설치



뿐만 아니라,  
우리의 응용을 위하여



CLOUD  
BARISTA



- ✓ 멀티클라우드 가상/공통 네트워크
- ✓ 멀티클라우드 인프라
- ✓ 글로벌 스케일 인프라

- ✓ 서로 다른 클라우드의 서로 다른 서브넷 상에서 인프라 및 응용(VM, Container 등)들이 **동일 서브넷에 존재하는 것처럼 사설 IP 기반으로 운용, 관리할 수 있도록 하는 기술**
- ✓ 멀티클라우드의 다양한 네트워크에 적응가능한 오버레이 네트워크로 VM 그룹에 동일 네트워크를 제공함

# Resolve, 멀티클라우드에서 발생하는 이슈

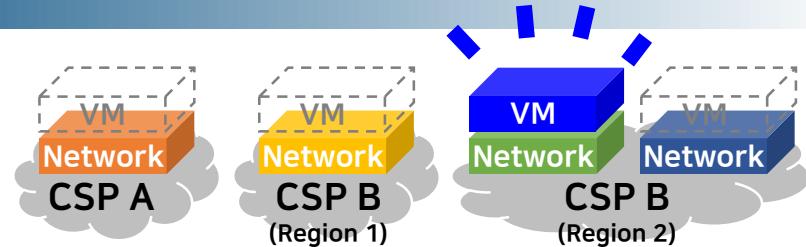
특히, 상이함(다양성) 및 가변성

## 멀티클라우드에서 발생하는 이슈 (자세한 내용은 부록을 참고해 주세요 ^o^)

### ✓ 유동성

클라우드 자원은 유동적이다?!

→ 인프라(MCIS) 생성 전에는 정확한 네트워크 정보를 얻기 어려워요.

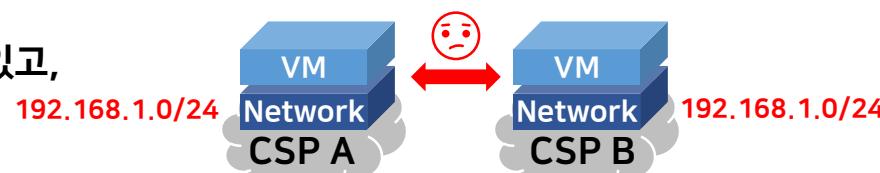


### ✓ 중복성

클라우드간에 중복된 Subnet이 생성 될 수 있다?!

→ VPN Gateway를 활용해도 통신이 어려울 수 있고, Supernetting\* 적용 또한 어려워요.

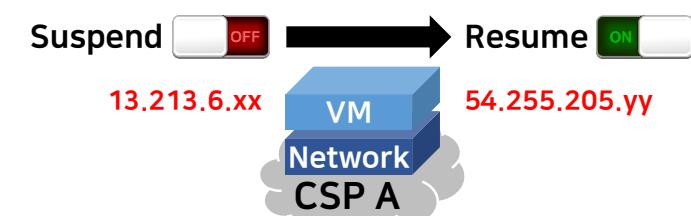
\* 여러 주소 영역을 하나의 주소 영역으로 통합하는 개념



### ✓ 변동성

VM관련 네트워크는 변경될 수 있다?!

→ 통신에 문제가 발생 할 수 있어요.  
(i.e., Suspend → Resume 아이피 주소 변경)



### ✓ 확장성

글로벌 스케일 인프라?!

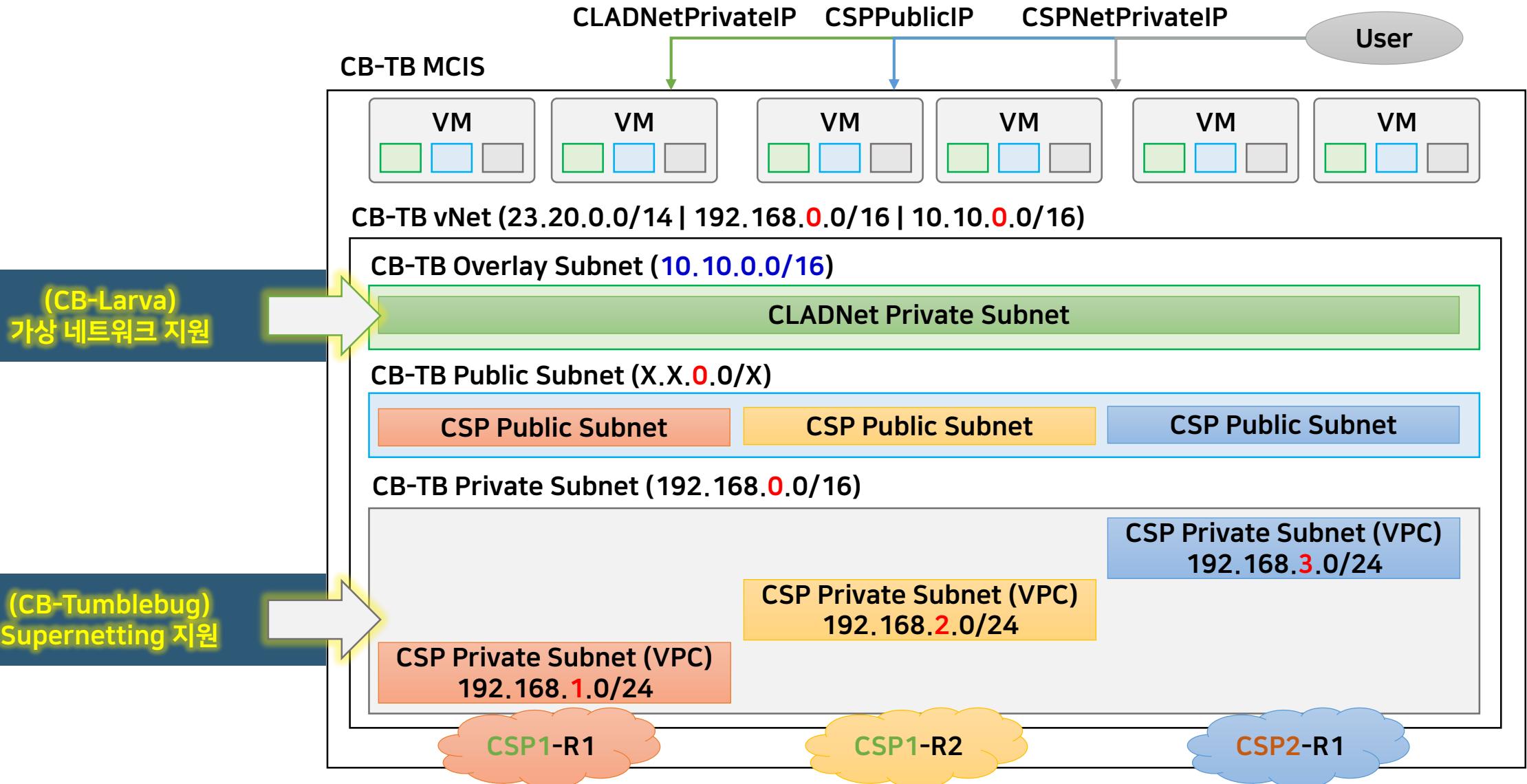
→ 적합한 규모의 주소공간(Address space)를 제공하기 어렵거나, 못할 수 있어요.



# MCIS 네트워크 구성 형상

MCIS: Multi-Cloud Infra Service

#우리는





# 또 하나의 “Adaptive”, cb-network service?!

gRPC



gRPC-Gateway



RESTful API  
OpenAPI Specification



## Protocol Documentation

### Table of Contents

- cloud\_barista\_network.proto
  - AvailableIPv4PrivateAddressSpaces
  - CLADNetRequest
  - CLADNetSpecification
  - CLADNetSpecifications
  - CloudInformation
  - ControlRequest
  - ControlResponse
  - DeletionResult
  - IPv4CIDRs
  - NetworkingRule
  - Peer
  - PeerRequest
  - Peers
  - TestRequest
  - TestResponse
  - UpdateDetailsRequest
  - CommandType
  - TestType
  - CloudAdaptiveNetworkService
  - SystemManagementService
  - Scalar Value Types

Official support		
These are the officially supported gRPC language, platform and OS versions:		
Language	OS	Compilers / SDK
C/C++	Linux, Mac	GCC 5.1+, Clang 4+
C/C++	Windows 7+	Visual Studio 2015+
C#	Linux, Mac	.NET Core, Mono 4+
C#	Windows 7+	.NET Core, NET 4.5+
Dart	Windows, Linux, Mac	Dart 2.12+
Go	Windows, Linux, Mac	Go 1.13+
Java	Windows, Linux, Mac	Java 8+ (KitKat+ for Android)
Kotlin	Windows, Linux, Mac	Kotlin 1.3+
Node.js	Windows, Linux, Mac	Node v8+
Objective-C	macOS 10.10+, iOS 9.0+	Xcode 7.2+
PHP	Linux, Mac	PHP 7.0+
Python	Windows, Linux, Mac	Python 3.5+
Ruby	Windows, Linux, Mac	Ruby 2.3+



Swagger  
[https://raw.githubusercontent.com/cloud-barista/cb-larva/main/poc-cb-net/docs/cloud\\_barista\\_network.swagger](https://raw.githubusercontent.com/cloud-barista/cb-larva/main/poc-cb-net/docs/cloud_barista_network.swagger) Explore

### Cloud-Barista Network (cb-network) service <sup>1.0</sup>

https://raw.githubusercontent.com/cloud-barista/cb-larva/main/poc-cb-net/docs/cloud\_barista\_network.swaggerjson

Note - [cloud\\_barista\\_network.swagger.json](#) is generated by [openapi2](#)

Cloud-Barista organization - Website  
Send email to Cloud-Barista organization  
Apache License Version 2.0

#### SystemManagementService

GET /health Checks service health

GET /v1/control/cladnet/{cladnetId}/command/{commandType} Controls a Cloud Adaptive Network from the remote

POST /v1/test/cladnet/{cladnetId}/type/{testType} Tests a Cloud Adaptive Network

#### CloudAdaptiveNetworkService

GET /v1/cladnet Get a list of Cloud Adaptive Network specifications

POST /v1/cladnet Create a new Cloud Adaptive Network

POST /v1/cladnet/availableIPv4AddressSpaces Recommend available IPv4 private address spaces for Cloud Adaptive Network

GET /v1/cladnet/{cladnetId} Get a Cloud Adaptive Network specification

DELETE /v1/cladnet/{cladnetId} [To be provided] Delete a Cloud Adaptive Network

PUT /v1/cladnet/{cladnetId} Update a Cloud Adaptive Network

GET /v1/cladnet/{cladnetId}/peer Get a list of peers in a Cloud Adaptive Network

GET /v1/cladnet/{cladnetId}/peer/{hostId} Get a peer in a Cloud Adaptive Network

PUT /v1/cladnet/{cladnetId}/peer/{hostId}/details Update a peer's details

GET /v1/cladnet/{cladnetId}/peer/{hostId}/networkingRule Get a networking rule of a peer

다양한

프로그래밍 언어

지원 가능

(하나의 Port만 사용중 입니다 ^^)



# cb-network를 입은(Clad) CB-Tumblebug

## CB-Tumblebug에 관련 API 추가 (under-development)

cb-network service 기반으로  
Cloud Adaptive Network(CLADNet)를 MCIS에 배치가능

This screenshot shows the Swagger UI interface for the Cloud-Barista Network (cb-network) service. It includes the URL [https://raw.githubusercontent.com/cloud-barista/cb-larva/main/poc-cb-net/docs/cloud\\_barista\\_network.swagger.json](https://raw.githubusercontent.com/cloud-barista/cb-larva/main/poc-cb-net/docs/cloud_barista_network.swagger.json) and an 'Explore' button. The interface lists various API endpoints for SystemManagementService and CloudAdaptiveNetworkService.

### Cloud-Barista Network (cb-network) service <sup>1.0</sup>

Note - [cloud\\_barista\\_network.swagger.json](#) is generated by [openapi2](#)

Cloud-Barista organization - Website  
Send email to Cloud-Barista organization  
Apache License Version 2.0

#### SystemManagementService

- `GET /health` Checks service health
- `GET /v1/control/cladnet/{cladnetId}/command/{commandType}` Controls a Cloud Adaptive Network from the remote
- `POST /v1/test/cladnet/{cladnetId}/type/{testType}` Tests a Cloud Adaptive Network

#### CloudAdaptiveNetworkService

- `GET /v1/cladnet` Get a list of Cloud Adaptive Network specifications
- `POST /v1/cladnet` Create a new Cloud Adaptive Network
- `POST /v1/cladnet/availableIPv4AddressSpaces` Recommend available IPv4 private address spaces for Cloud Adaptive Network
- `GET /v1/cladnet/{cladnetId}` Get a Cloud Adaptive Network specification
- `DELETE /v1/cladnet/{cladnetId}` [To be provided] Delete a Cloud Adaptive Network
- `PUT /v1/cladnet/{cladnetId}` Update a Cloud Adaptive Network
- `GET /v1/cladnet/{cladnetId}/peer` Get a list of peers in a Cloud Adaptive Network
- `GET /v1/cladnet/{cladnetId}/peer/{hostId}` Get a peer in a Cloud Adaptive Network
- `PUT /v1/cladnet/{cladnetId}/peer/{hostId}/details` Update a peer's details
- `GET /v1/cladnet/{cladnetId}/peer/{hostId}/networkingRule` Get a networking rule of a peer



This screenshot shows the Swagger UI interface for the CB-Tumblebug REST API. It includes the URL [Base URL: /tumblebug](#), a 'doc.json' link, and an 'Explore' button. The interface features a cartoon illustration of a yellow beetle with a red 'N' on its shell, standing on a red base with a circular pattern. It lists three main sections: [Infra service] MCIS Resource monitor (for developer), [Infra service] MCIS Cloud Adaptive Network (for developer), and [Infra service] MCIS Auto control policy management (WIP). The third section is highlighted with a blue border. Each section contains several API endpoints with detailed descriptions and examples.

**CB-Tumblebug REST API <sup>latest</sup>**  
[ Base URL: /tumblebug ]  
[doc.json](#)

**CB-Tumblebug REST API**  
[API Support - Website](#)  
[Send email to API Support](#)  
[Apache 2.0](#)

**[Infra service] MCIS Resource monitor (for developer)**

`POST /ns/{nsId}/monitoring/install/mcis/{mcisId}` Install monitoring agent (CB-Dragonfly agent) to MCIS

`GET /ns/{nsId}/monitoring/mcis/{mcisId}/metric/{metric}` Get monitoring data of specified MCIS for specified monitoring metric (cpu, memory, disk, network)

**[Infra service] MCIS Cloud Adaptive Network (for developer)**

`PUT /ns/{nsId}/network/mcis/{mcisId}` Inject Cloud Information For Cloud Adaptive Network

`POST /ns/{nsId}/network/mcis/{mcisId}` Configure Cloud Adaptive Network (cb-network agent) to MCIS

**[Infra service] MCIS Auto control policy management (WIP)**

`GET /ns/{nsId}/policy/mcis` List all MCIS policies

`DELETE /ns/{nsId}/policy/mcis` Delete all MCIS policies

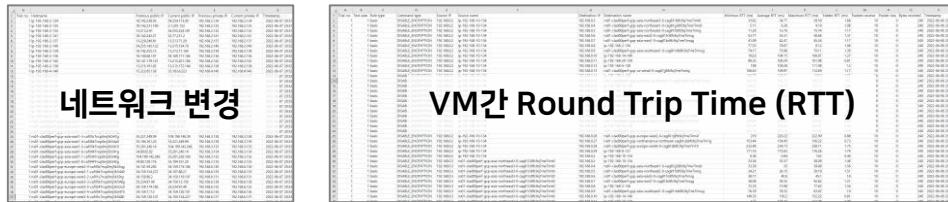
`GET /ns/{nsId}/policy/mcis/{mcisId}` Get MCIS Policy

`POST /ns/{nsId}/policy/mcis/{mcisId}` Create MCIS Automation policy

`DELETE /ns/{nsId}/policy/mcis/{mcisId}` Delete MCIS Policy

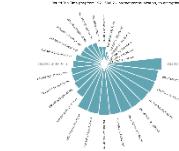
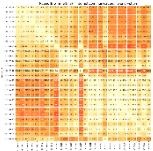
# cb-network 시스템 +

## + perf-eval client



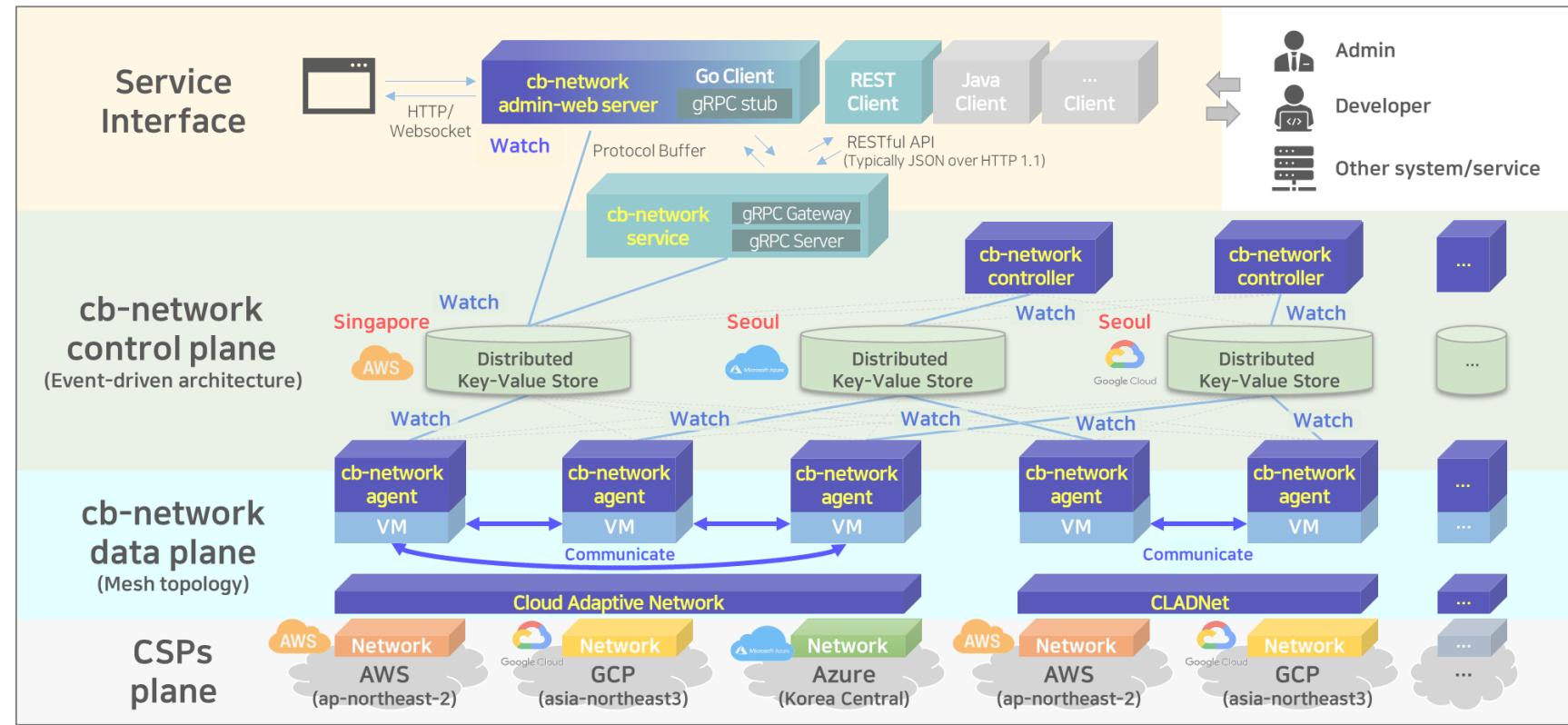
## + cb-network visualizer

RTT →



Heatmap

Circular barplot



cb-network 시스템의 세부 구조 및 컴포넌트

#구조 #POC수행중

**+ End-to-end encryption**  
 "secret-util"제공(Go package)  
 (공개SW) 강력한 보안을 위해 기여 바랍니다^^;

**Cost**

**Price**



멀티클라우드 인프라 및 응용을 위한 네트워크

(함께 고민해볼 포인트 일까요? ^^;;)



# 클라우드의 주요 과금 대상

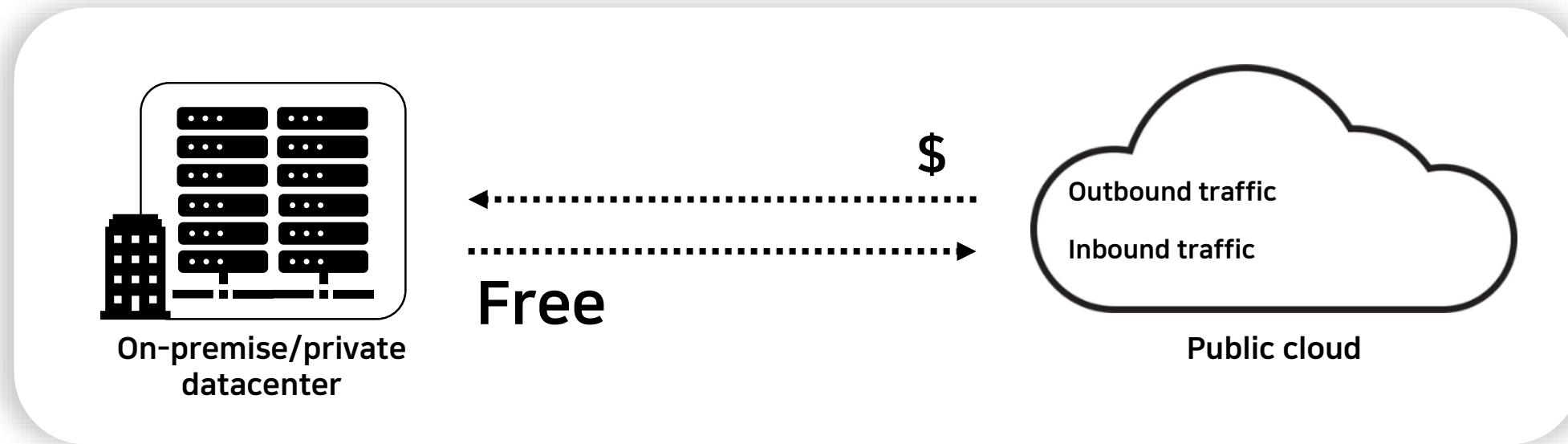
---

✓ Compute

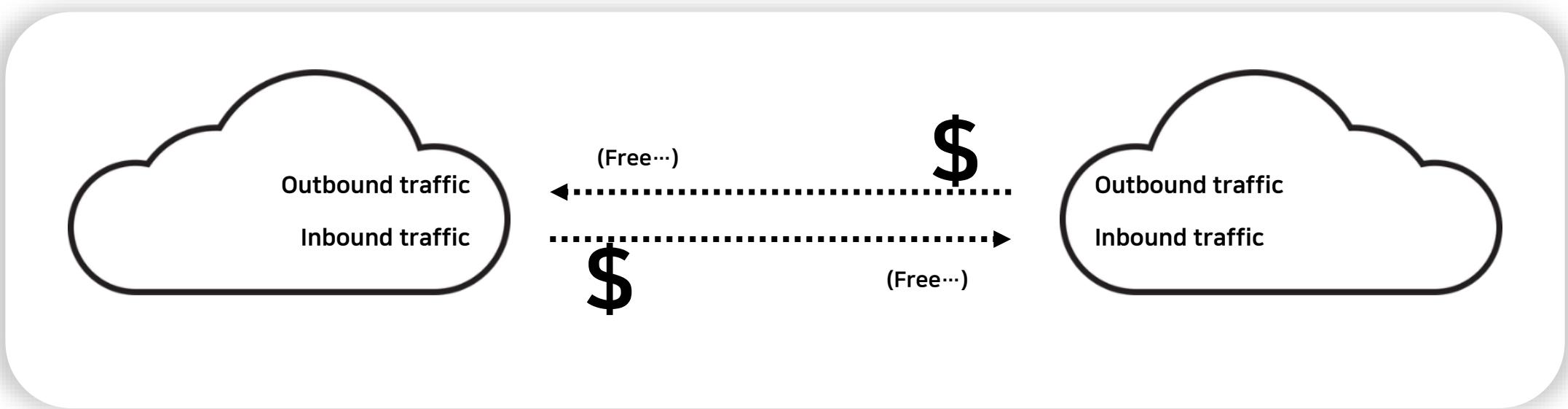
✓ Network

✓ Storage

(주로 다를 과금 대상)



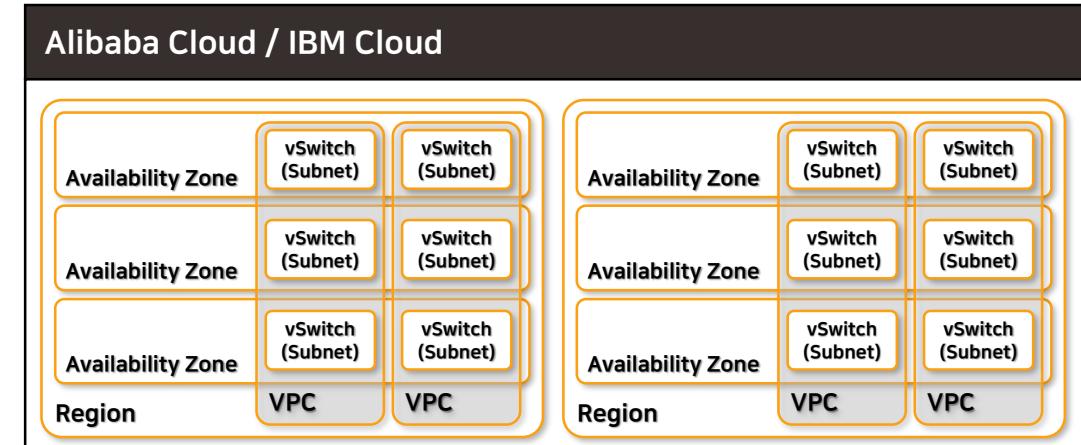
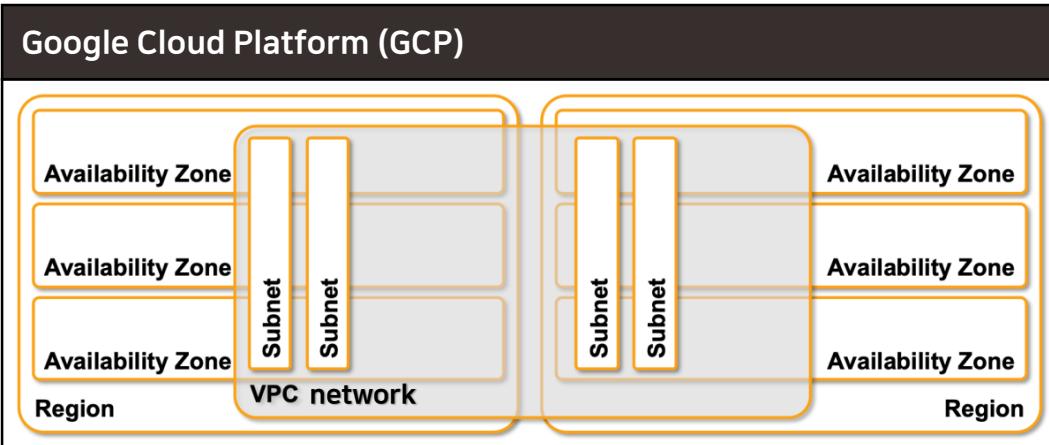
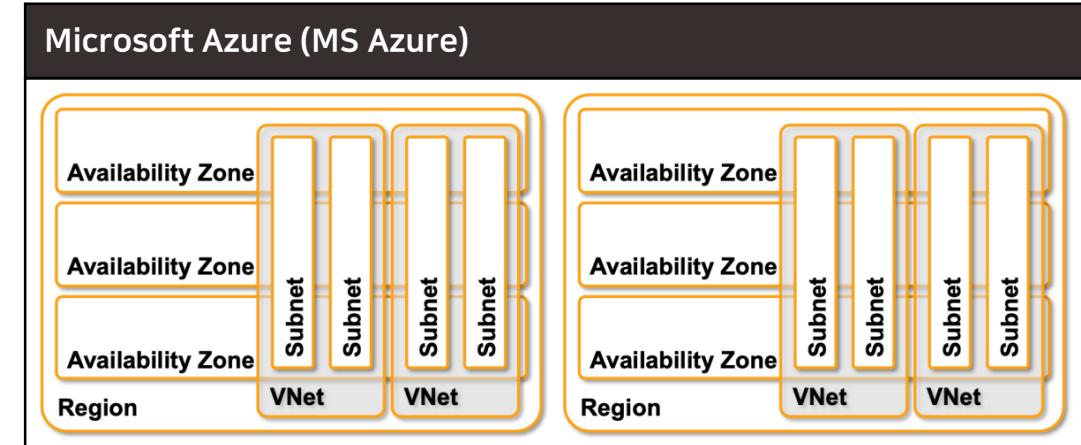
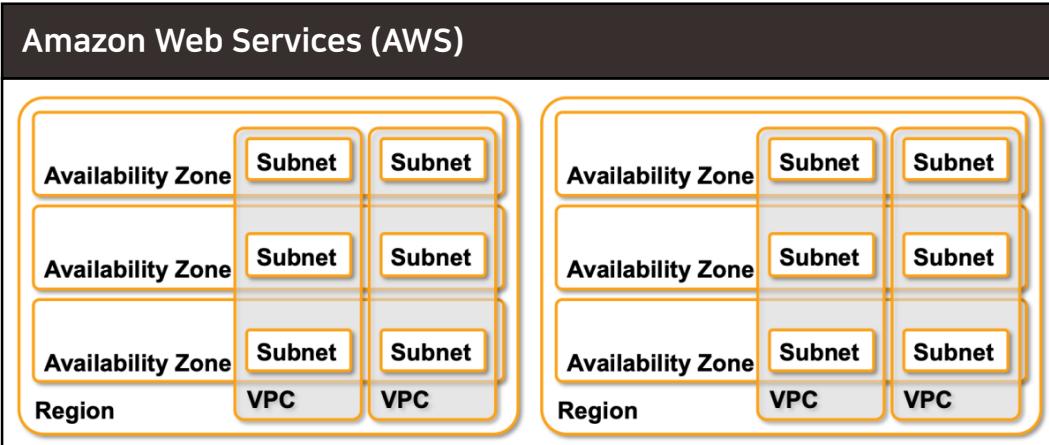
42% of enterprises use the cloud  
for backup



# CSP의 네트워크 형상 및 관계 분석

구역 구분 참고, 주관적임:  
 (같음) 물리적 구역 - Region, Availability Zone  
 (다름) 논리적 구역 - VPC/vNet/VPC network, Subnet/vSwitch

\* 주요 포인트: Regions, availability zones, virtual networks, and subnets의 격리 범위 및 관계



출처:

ipSpace.net, "Virtual Networks and Subnets in AWS, Azure, and GCP", 2021 (accessed on 2022-01-25 <https://blog.ipspace.net/2021/02/vpc-subnets-aws-azure-gcp.html>)

ipSpace.net, "Availability Zones and Regions in AWS, Azure and GCP", 2021 (accessed on 2022-01-25 <https://blog.ipspace.net/2021/02/public-cloud-regions-availability-zones.html>)

Mate Gulic, "AWS, Azure, GCP: Virtual Networking Concepts overview", 2020 (accessed on 2022-01-25 <https://www.linkedin.com/pulse/aws-azure-gcp-virtual-networking-concepts-overview-mate-gulic/>)

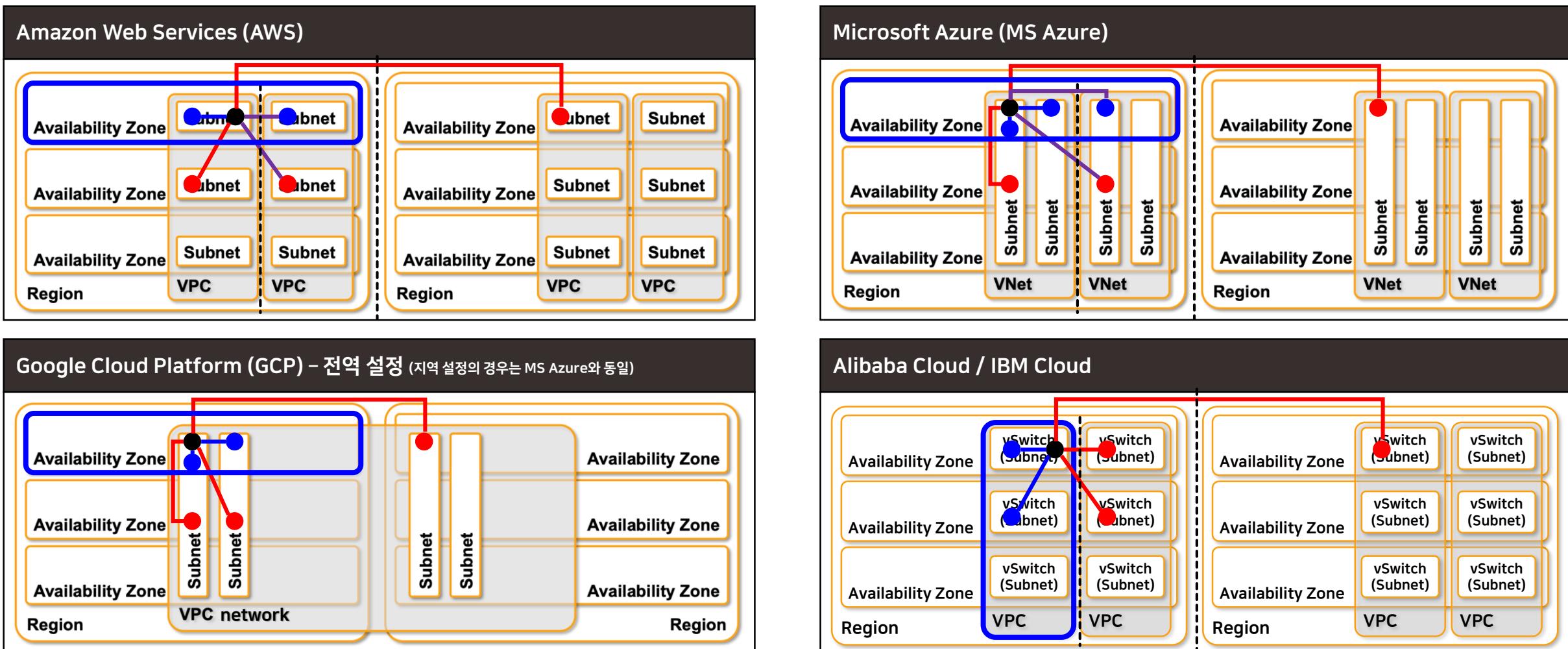
Alibaba Cloud, "Overview – VPCs and vSwitches", (accessed on 2022-01-26 <https://www.alibabacloud.com/help/en/doc-detail/100380.htm>)

IBM Cloud, "About networking", 2020 (accessed on 2022-01-26 <https://cloud.ibm.com/docs/vpc?topic=vpc-about-networking-for-vpc>)

Tencent Cloud, "Virtual Private Cloud Product Introduction Product Documentation", 2020. (accessed on 2022-01-26 [https://main.qcloudimg.com/raw/document/intl/product/pdf/215\\_532\\_en.pdf](https://main.qcloudimg.com/raw/document/intl/product/pdf/215_532_en.pdf))

# Data transfer billing 고찰

구역 구분 참고, 주관적임:  
 (같음) 물리적 구역 - Region, Availability Zone  
 (다름) 논리적 구역 - VPC/vNet/VPC network, Subnet/vSwitch



[구분] 파란색 선: free of charge / 빨간색 선: charged / 보라색 선: could be charged, 다른 요인으로 인한 요금 발생 가능 (data transfer billing (X))  
 예) Virtual Private Network (VPN), Transit Gateway, NAT Gateway 등

# (사례) Snowflake

멀티클라우드 과금 체계를 잘 파고든 사례 ?

## MARKETS

### Warren Buffett's Berkshire Hathaway just made a fast \$800 million on Snowflake's surging IPO

PUBLISHED WED, SEP 16 2020-1:07 PM EDT | UPDATED WED, SEP 16 2020-6:06 PM EDT



Yun Li  
@YUNLI026



Maggie Fitzgerald  
@MKMFITZGERALD

WATCH LIVE

## KEY POINTS

- Berkshire Hathaway just made about \$800 million off of Snowflake's market debut, despite Warren Buffett's aversion to IPOs.
- It's widely speculated that Buffett lieutenants Todd Combs and Ted Weschler orchestrated the Snowflake bet.
- The legendary value investor hasn't invested in a newly public company since the Ford IPO in 1956.



Warren Buffett  
Gerald Miller/CNBC

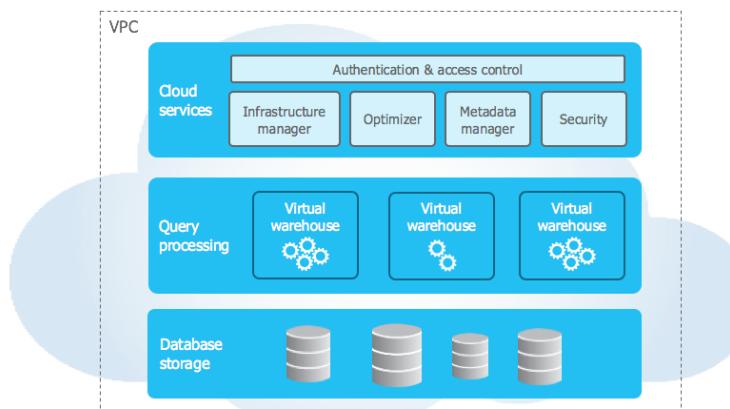
Source: CNBC

# (참고) Snowflake 개요

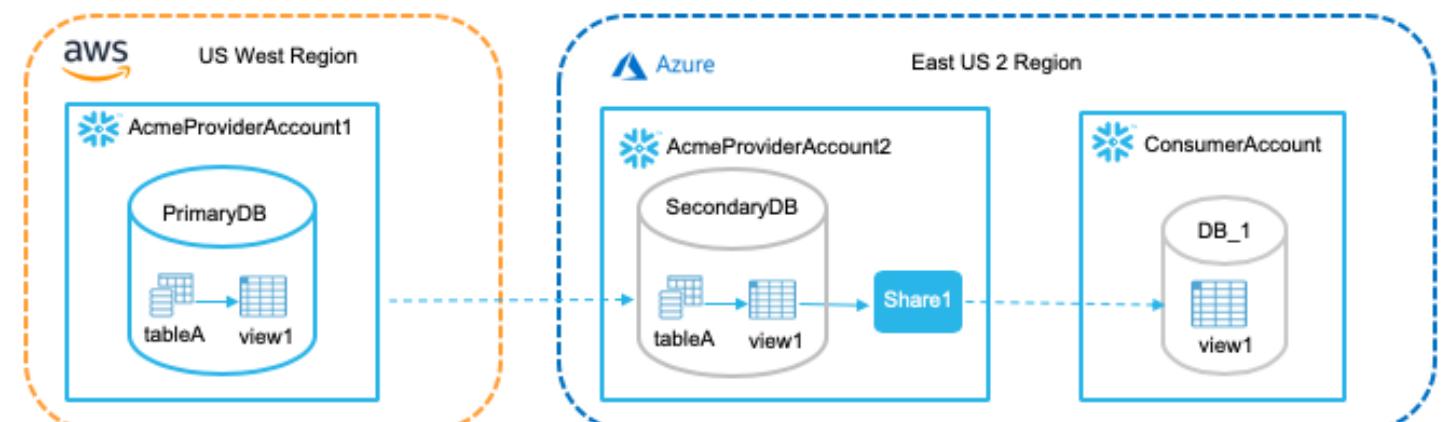
## snowflake® : Data Platform as a Cloud Service

Snowflake is a true SaaS offering. More specifically:

- There is no hardware (virtual or physical) to select, install, configure, or manage.
- There is virtually no software to install, configure, or manage.
- Ongoing maintenance, management, upgrades, and tuning are handled by Snowflake.



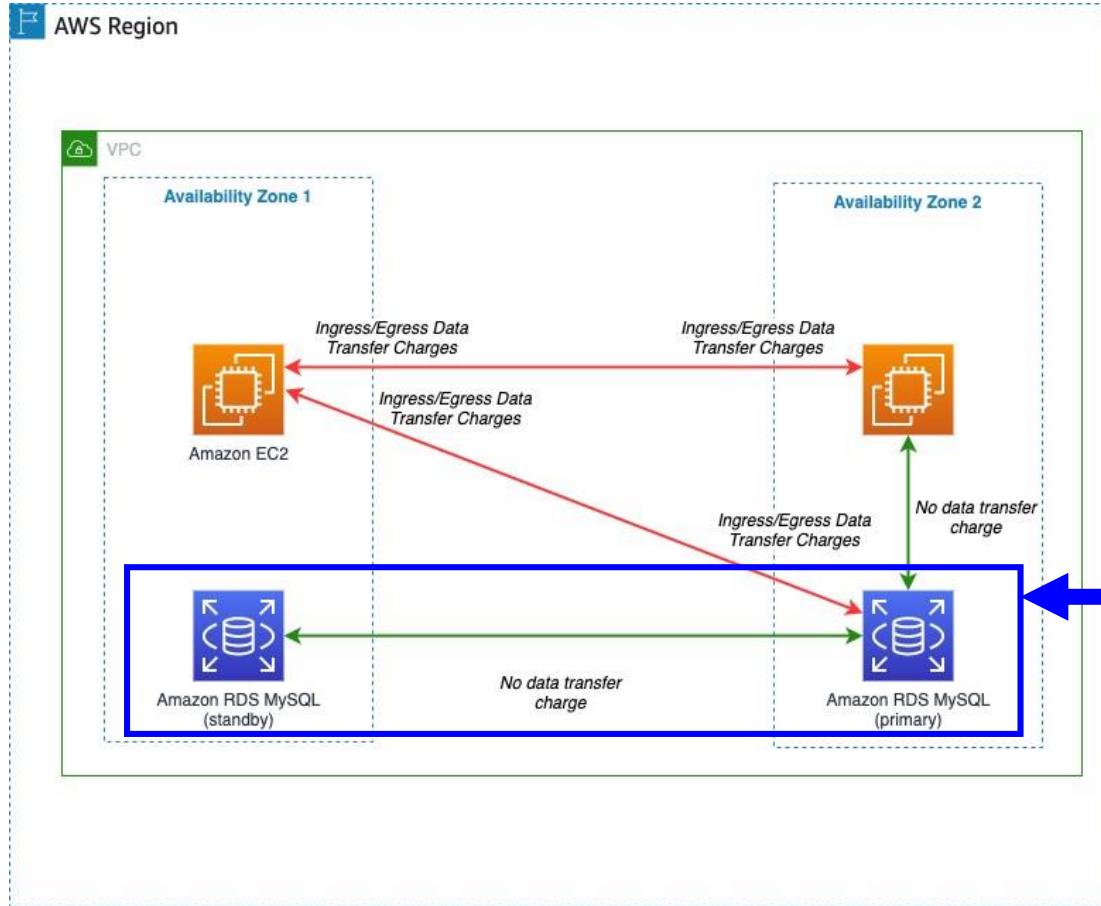
Snowflake Architecture



(Example 1) A data provider, Acme, wants to share data with data consumers in a different region.

# AWS 과금 구간 ← 연관성 → Snowflake의 데이터 공유 및 제공 사례

## 동일 AWS Region 상에서 Data Transfer Billing



Availability Zone이 다른에도

No data transfer charge

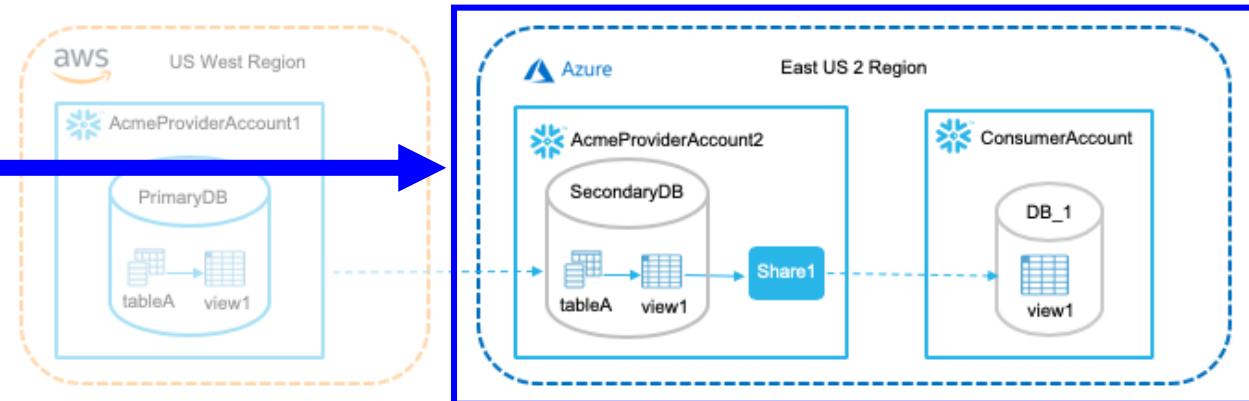


Figure 3. Workload components across Availability Zones

출처:

AWS, "Overview of Data Transfer Costs for Common Architectures", 2021 (accessed on 2022-01-25 <https://aws.amazon.com/ko/blogs/architecture/overview-of-data-transfer-costs-for-common-architectures/>)

# 멀티클라우드 관점의 시스템/SW 아키텍처링 필요

#의견 #제안

멀티클라우드를 기반으로/타겟으로 하는 신규 서비스라면?

→ 멀티클라우드 특성을 반영한 시스템/SW 아키텍처링을 고려해 보는게 어떨까요? ^^

(이런 느낌 일까요? ^^;;)

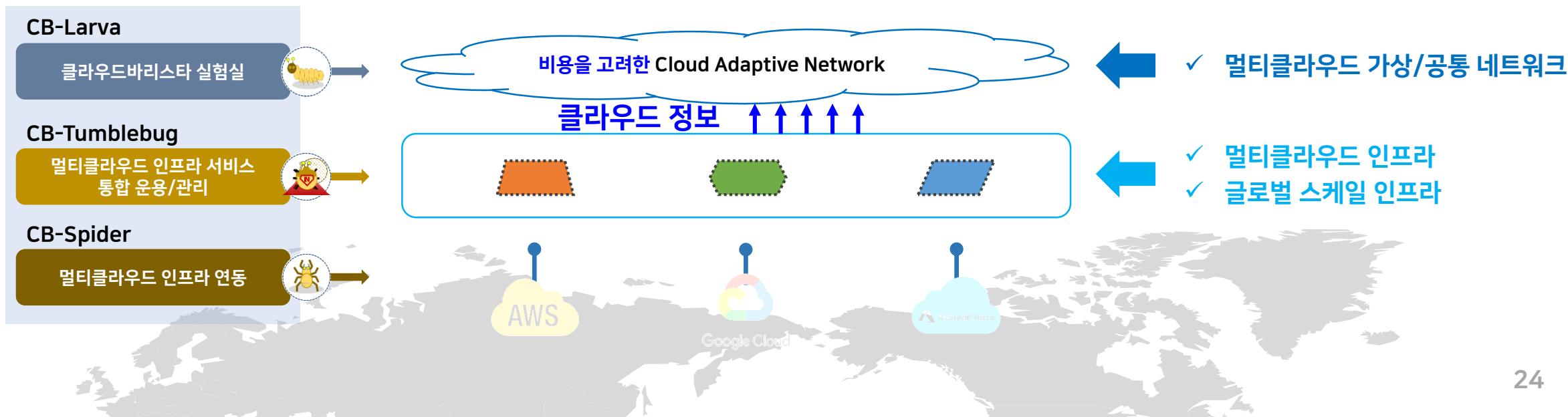
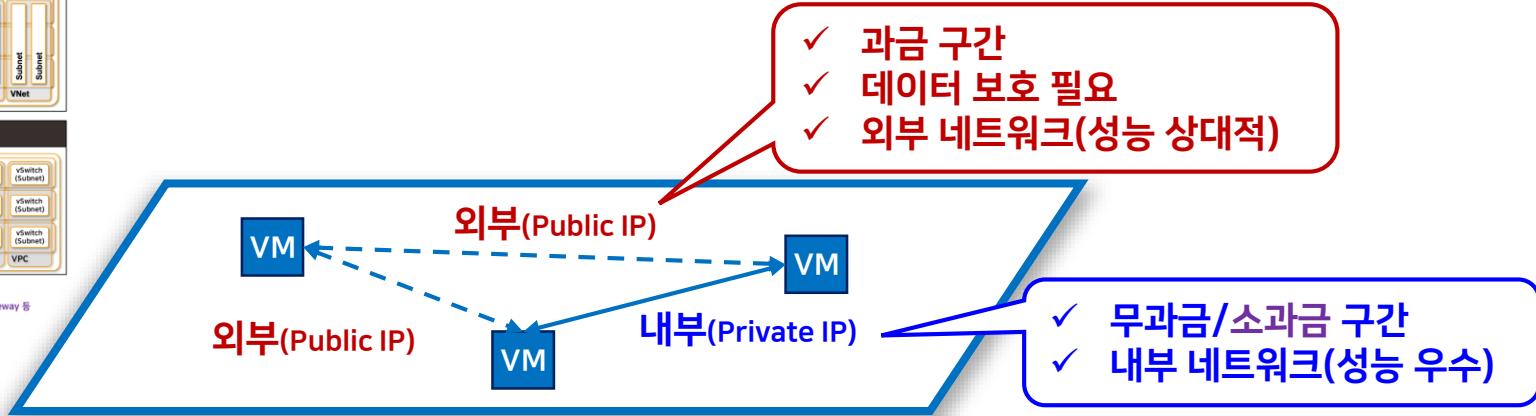
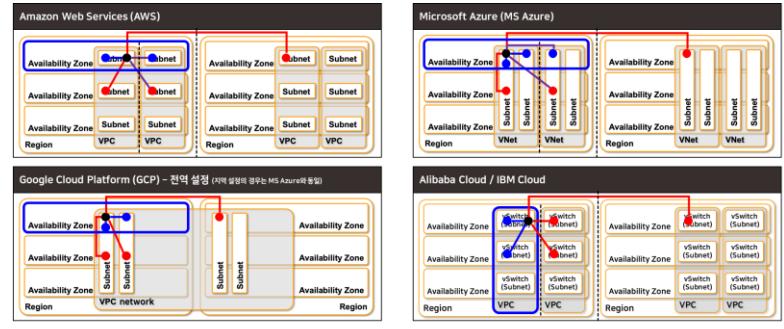
“멀티”클라우드 네이티브 서비스?! ← 클라우드 네이티브 서비스 + 멀티클라우드의 과금 체계를 겹들여… ;;

(멀티클라우드 네트워크를 위한 시스템/SW 개선)

비용을 고려한 Cloud Adaptive Network → 멀티클라우드 인프라 및 응용을 위한 네트워크

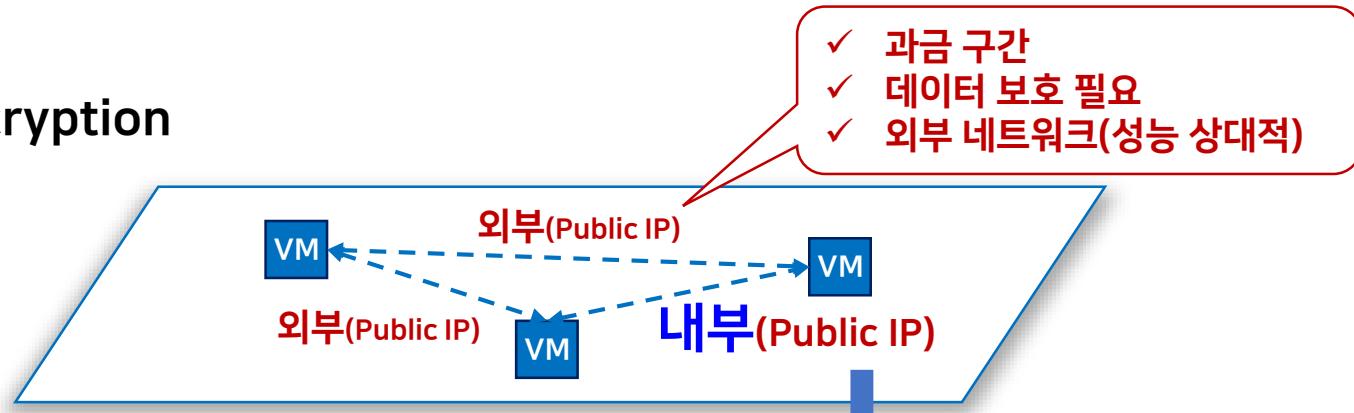
- + (Idea) 사용자 측면에서 바라보면 보이는 것
  - 1. 비용(Cost): 성능(performance), 과금(price)
  - 2. 사용자의 자원을 우선 고려 → 무과금 구간 판단 용이

# (환경소개) 비용을 고려한 Cloud Adaptive Network

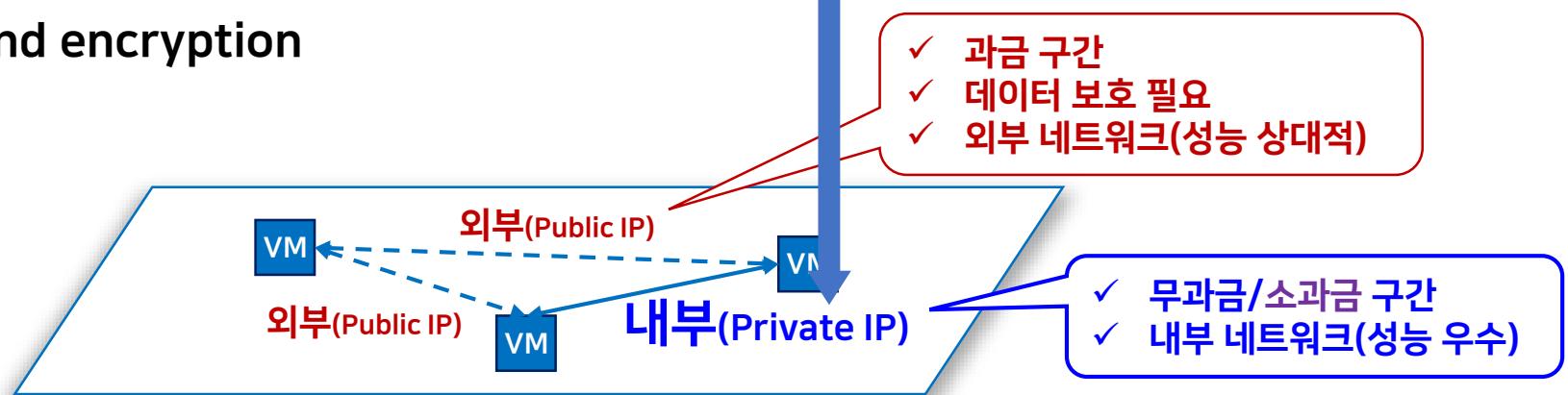


# 실험 시나리오

1. 인터넷 통신(Public IP)
2. 인터넷 통신 + End-to-end encryption



3. 비용을 고려한 통신(선택, Private IP/Public IP)
4. 비용을 고려한 통신 + End-to-end encryption



\* RTT: Round Trip Time

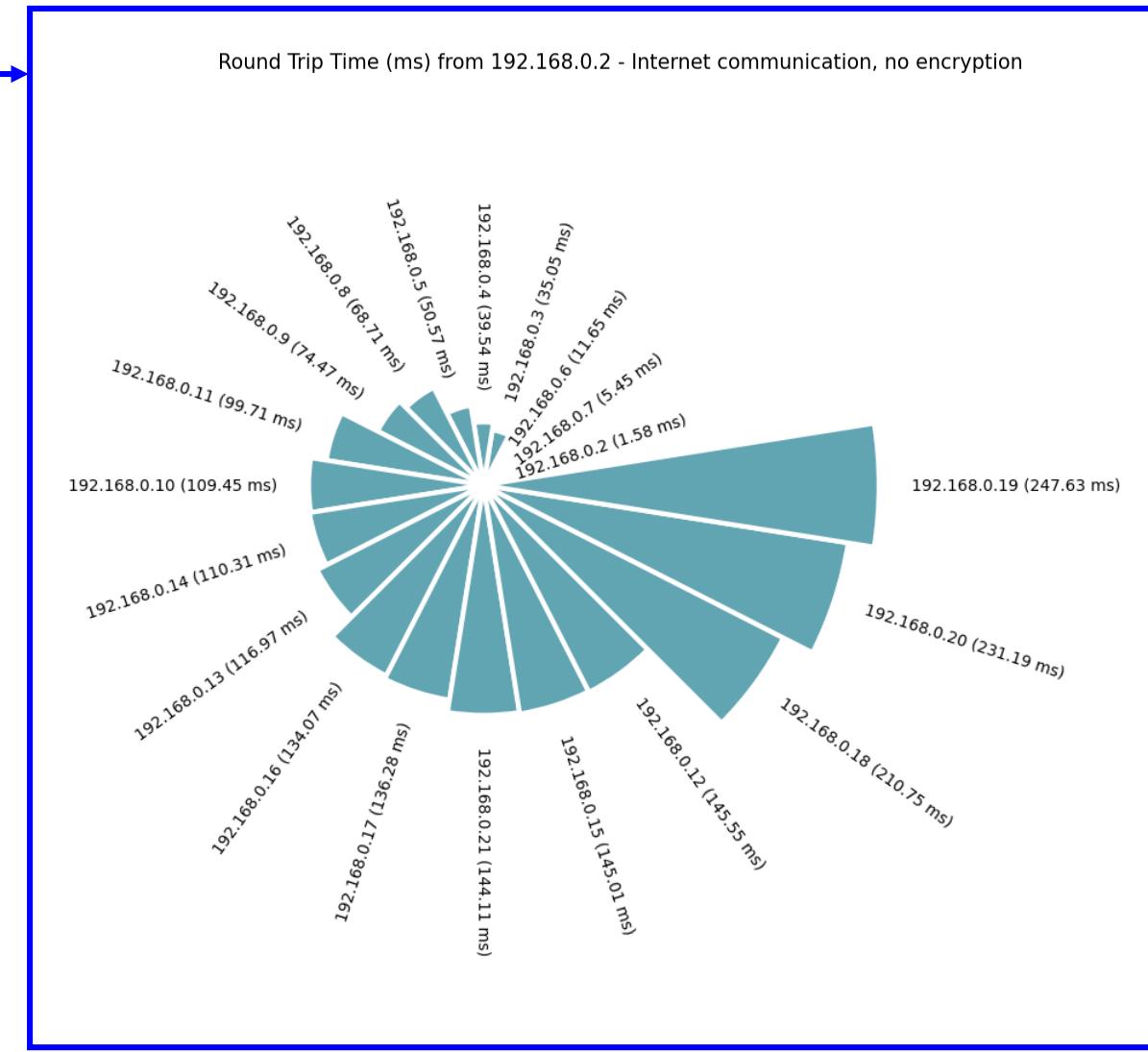


# (맛보기) RTT between 20 Regions in 2 CSPs (전세계 클라우드의 Region간 성능테스트 가능)

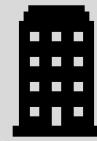
\* A: AWS, G: GCP

Round Trip Time (ms) - Internet communication, no encryption

	192.168.0.2 -	1.6	35.0	39.5	50.6	11.7	5.5	68.7	74.5	109.5	99.7	145.6	117.0	110.3	145.0	134.1	136.3	210.8	247.6	231.2	144.1
A	192.168.0.3 -	34.8	0.5	39.3	39.3	36.9	35.2	78.9	71.7	140.9	130.6	150.7	150.3	136.6	193.0	183.2	127.2	245.6	278.9	255.9	178.9
G	192.168.0.4 -	39.6	39.5	0.4	15.4	35.5	38.6	52.2	51.8	142.8	134.0	140.9	145.0	141.9	192.1	184.5	105.7	247.9	278.1	257.1	183.1
G	192.168.0.5 -	50.9	38.9	14.9	0.5	45.5	51.7	39.1	38.3	154.5	147.8	136.2	130.6	152.2	203.5	196.0	96.9	242.5	258.1	219.3	195.1
G	192.168.0.6 -	11.6	36.0	35.3	46.0	0.7	11.0	83.2	82.7	122.0	112.6	130.7	127.8	117.0	171.8	163.8	133.9	227.5	258.3	238.1	163.7
G	192.168.0.7 -	4.7	34.9	38.7	52.2	11.4	0.5	71.2	71.3	114.7	107.3	123.0	113.8	108.1	165.6	157.6	128.4	218.4	248.5	231.0	156.4
A	192.168.0.8 -	68.6	78.5	51.5	39.5	82.6	71.0	0.4	4.8	176.1	170.8	95.8	95.5	172.0	209.7	199.1	67.3	168.6	225.9	200.3	211.0
G	192.168.0.9 -	74.0	70.7	51.2	37.3	82.3	70.0	3.8	0.3	178.3	170.4	95.7	95.7	171.0	225.2	219.2	75.1	208.0	224.7	194.1	219.2
A	192.168.0.10 -	109.5	140.9	142.2	154.9	122.4	114.7	176.0	180.1	0.6	24.0	147.8	140.2	19.3	81.4	54.9	234.8	149.4	173.9	154.6	64.1
A	192.168.0.11 -	100.1	132.3	135.2	149.4	114.6	110.8	174.4	172.6	26.4	1.0	157.4	142.0	40.6	62.5	51.5	230.1	139.7	188.2	168.6	71.7
G	192.168.0.12 -	145.3	150.6	140.3	135.5	131.3	123.1	96.2	96.6	147.3	156.9	0.6	4.4	146.1	209.6	202.6	155.3	265.5	295.6	275.7	201.5
A	192.168.0.13 -	117.2	151.1	145.5	131.2	128.5	115.1	95.8	96.6	139.8	141.7	5.0	1.0	148.9	199.1	188.7	151.6	265.1	305.4	285.5	199.9
G	192.168.0.14 -	110.0	136.3	141.3	152.0	118.0	108.5	172.2	171.5	18.5	39.2	146.8	148.8	0.5	74.3	64.0	247.3	135.6	165.1	145.5	62.9
A	192.168.0.15 -	144.4	192.7	192.0	203.7	172.6	166.1	210.4	226.1	80.4	63.3	210.4	199.7	74.4	0.8	27.5	191.2	81.0	121.0	91.6	17.2
A	192.168.0.16 -	133.8	183.1	184.4	196.4	164.2	158.1	200.1	220.0	55.2	52.0	203.1	188.8	65.7	28.1	1.3	199.8	88.9	118.1	101.3	15.3
A	192.168.0.17 -	135.9	127.1	105.6	96.4	134.4	128.9	68.0	75.4	233.9	229.7	155.3	151.3	247.8	190.1	198.4	0.8	114.9	166.6	132.9	189.1
A	192.168.0.18 -	212.3	247.1	249.9	242.8	229.2	218.9	170.3	209.2	149.4	139.7	267.4	265.6	137.0	82.3	89.1	116.0	0.3	44.5	19.3	78.8
G	192.168.0.19 -	247.7	277.9	278.6	258.2	259.2	248.3	226.6	224.7	173.3	186.6	295.9	305.0	165.5	119.0	116.8	166.1	42.1	0.5	34.6	107.6
G	192.168.0.20 -	231.4	255.6	257.7	219.0	237.7	230.6	199.6	193.8	153.9	167.1	274.7	284.5	145.9	90.8	99.8	132.6	17.3	34.5	0.5	89.3
A	192.168.0.21 -	144.8	179.4	182.9	195.3	164.7	156.3	212.4	220.2	64.3	70.8	202.3	198.9	62.8	17.4	16.1	189.5	78.1	107.8	90.6	0.8



# (가정) 사업 확장 사례



국내에서 잘나가는 A사가  
사업영역을 아시아로 확장

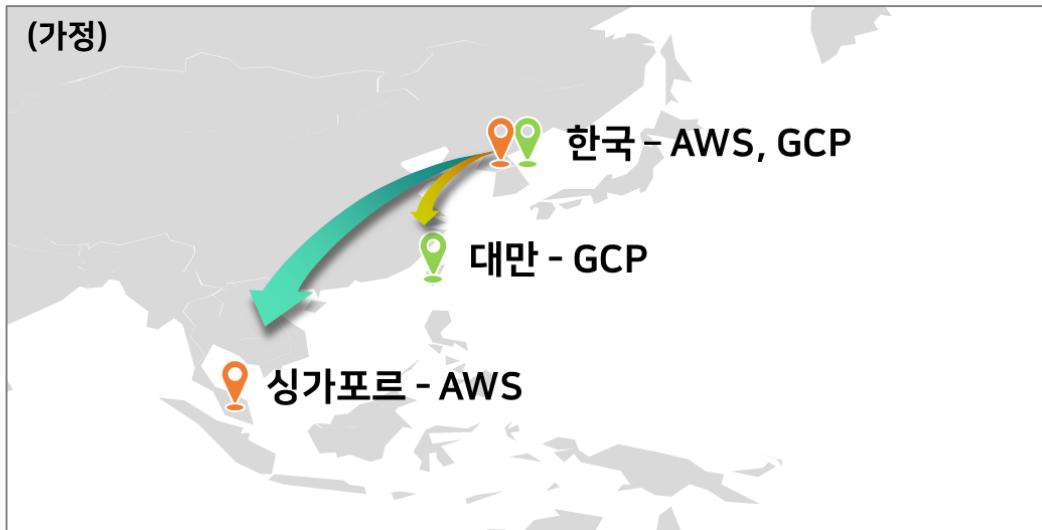


그 자체가 글로벌 스케일인, 멀티클라우드를 기반으로  
우리의 서비스를 세계 곳곳으로 보내는 그날까지…

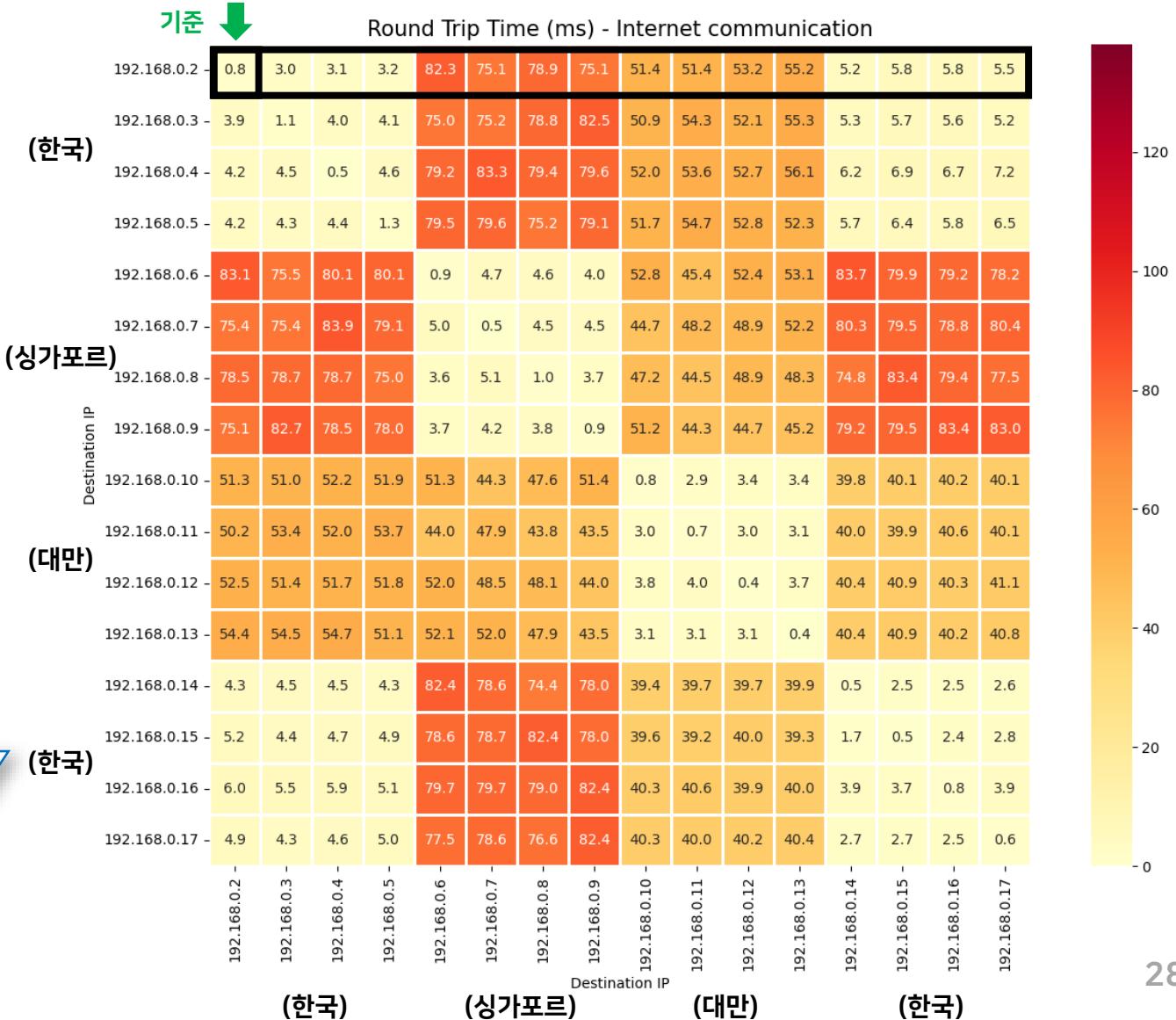
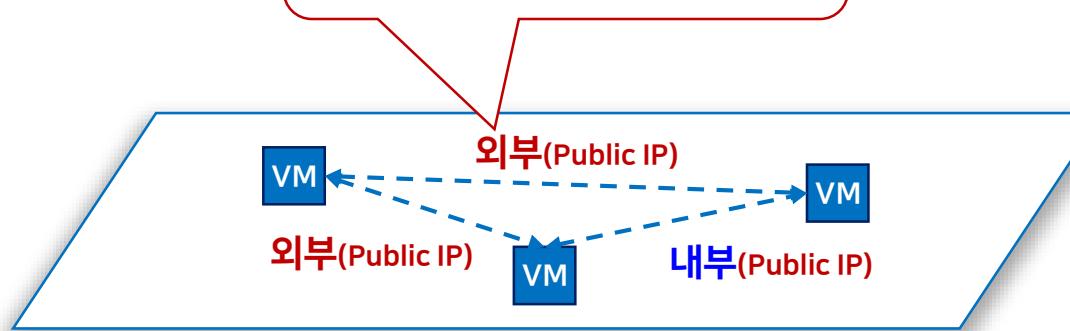
Cloud-Barista Community



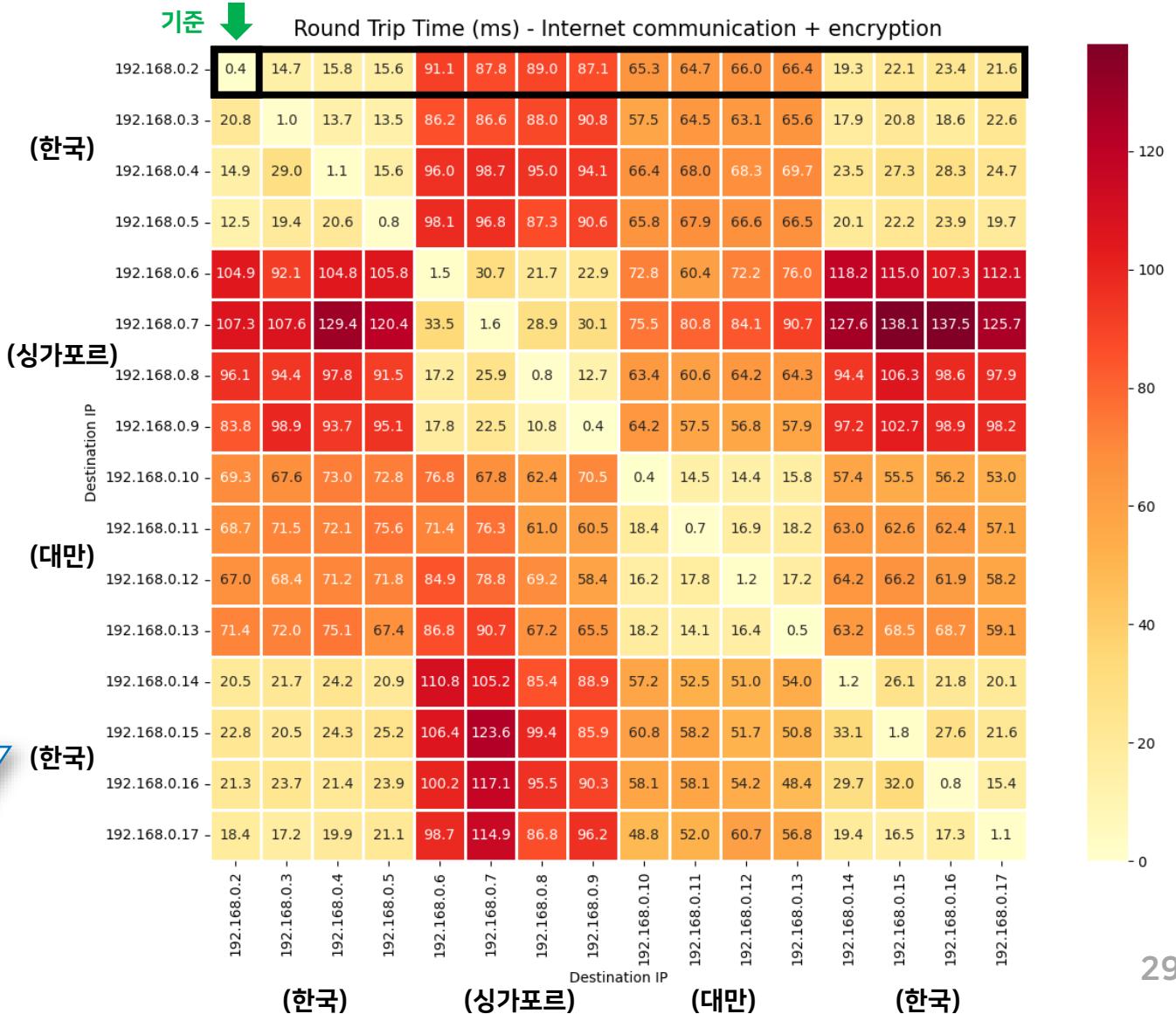
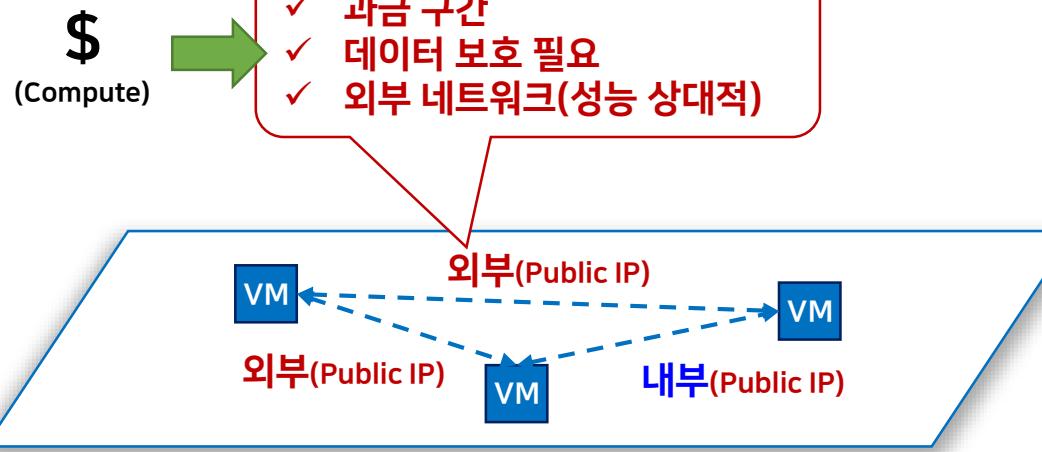
# 시나리오 1: 인터넷 통신(Public IP)



- ✓ 과금 구간
- ✓ 데이터 보호 필요
- ✓ 외부 네트워크(성능 상대적)



# 시나리오 2: 인터넷 통신 + End-to-end encryption



# 시나리오 3: 비용을 고려한 통신(선택, Private IP/Public IP)

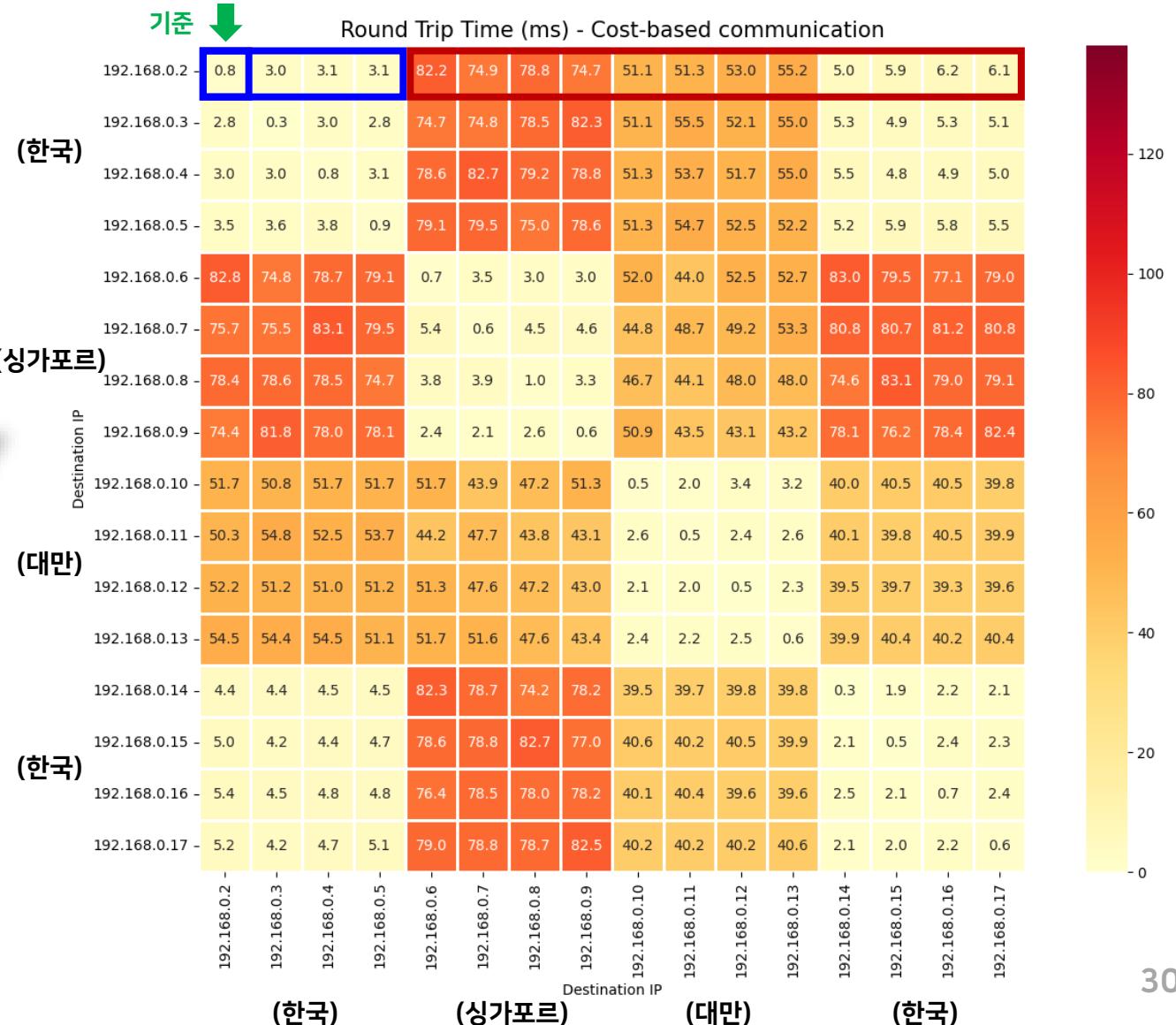
## 과금 구간을 기준으로 영역 구분 및 IP 선택 적용

- ✓ 과금 구간
- ✓ 데이터 보호 필요
- ✓ 외부 네트워크(성능 상대적)



- ✓ 무과금/소과금 구간
- ✓ 내부 네트워크(성능 우수)

(작지만, Internet Gateway(IGW) 관련해서 성능 이점이 아주 조금 있어요 ^^;;)



# 시나리오 4: 비용을 고려한 통신 + End-to-end encryption

절약!

\$ with 과금 구간 통신 최소화  
(Network)

(설계상/배치시)

- ✓ 과금 구간
- ✓ 데이터 보호 필요
- ✓ 외부 네트워크(성능 상대적)



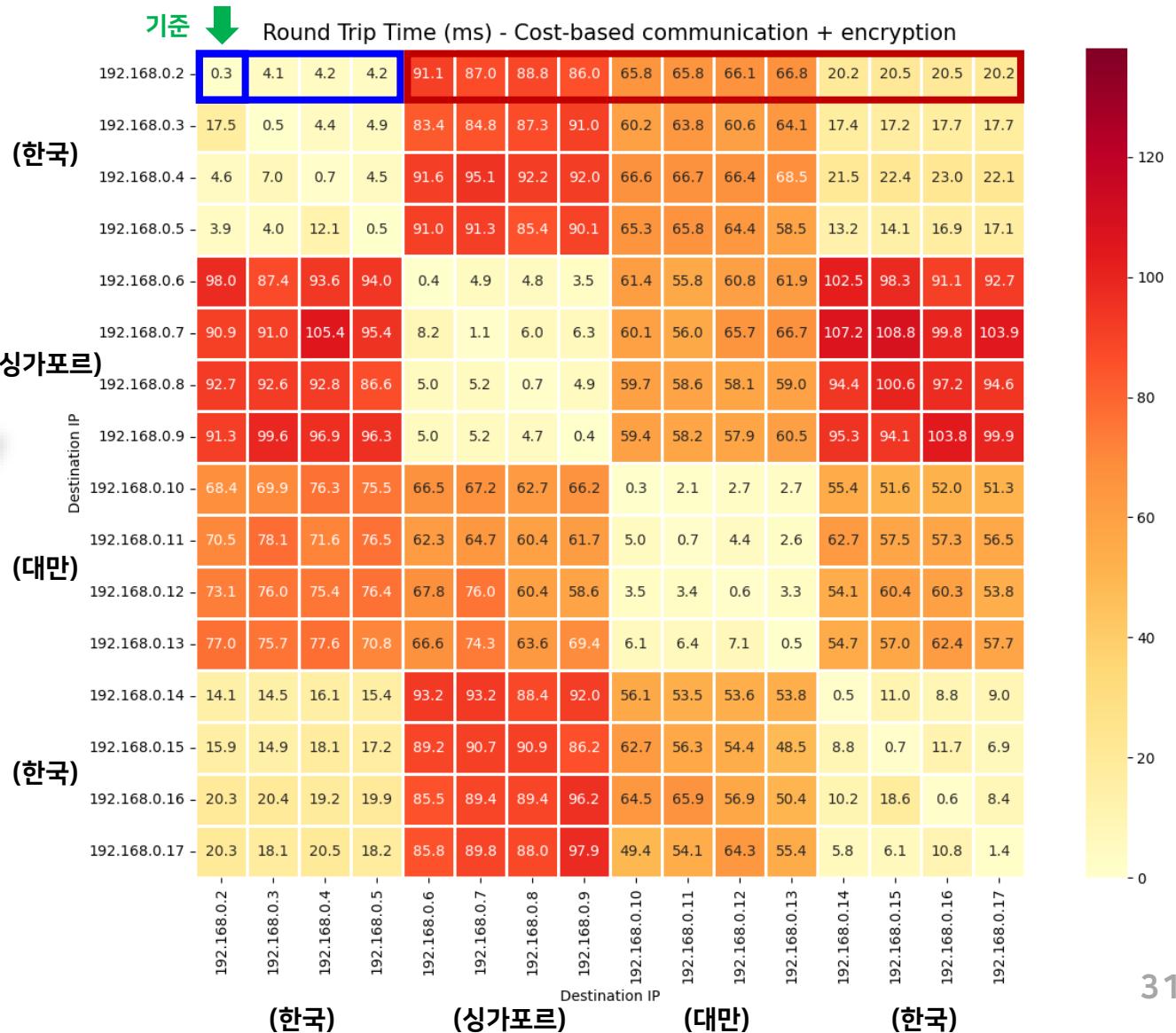
- ✓ 무과금/소과금 구간
- ✓ 내부 네트워크(성능 우수)
- ✓ 선택적 데이터 보호

절약!

\$  
(Compute)

기준

Round Trip Time (ms) - Cost-based communication + encryption



(예고)

## 실제 과금/비용 절약

다음 컨퍼런스를 기대하세요 ^^;;;

(측정 전에 해결해야 할 이슈가…)

- (같이 생각해볼 부분) 글로벌 클라우드의 국내 시장 점유율 from IDC
- ✓ 인프라 서비스 시장 점유율 약 78% 이상
  - ✓ 전반적인 클라우드 서비스 시장 점유율 약 47% 이상



# cb-network 컴포넌트 설치 및 구동 절차

#연구개발중  
#점점진화중

## 가이드: 소스코드 기반 설치 및 구동

### Install based on source code

Yunkon (Alvin) Kim edited this page on 13 Apr · 2 revisions

#### 설치 가이드

cb-network system의 컴포넌트를 설치 및 구동하기 위한 가이드입니다. (버전: v0.0.12)

cb-network system은 네트워크 제어 영역(Control plane)과 데이터 영역(Data plane)으로 구성되어 있습니다. 따라서, 이후 내용은 Control plane과 Data plane으로 나누어 설명하겠습니다.

- Control plane 컴포넌트:
  - cb-network controller,
  - cb-network service,
  - cb-network admin-web
- Data plane 컴포넌트:
  - cb-network agent

비고 - 각 컴포넌트는 독립된 노드(PC, VM, 등)에서 구동 가능합니다. 본 가이드에서는 복잡한 설명을 생략하고자 Control plane 컴포넌트를 한 노드에 구성했습니다.

#### 필요조건(Requirements)

cb-network system의 각 컴포넌트를 설치하기 전에, 아래 사항을 참고하십시오.

##### 1. 지원 플랫폼(Supported platforms)

Architecture	Operating system	Distribution	Version	Note

<https://github.com/cloud-barista/cb-larva/wiki/Install-based-on-source-code>

## 가이드: 컨테이너 기반 설치 및 구동 (docker-compose)

### Install based on container

Yunkon (Alvin) Kim edited this page 4 days ago · 2 revisions

#### 컨테이너 기반 설치 가이드

cb-network system의 컴포넌트를 설치 및 구동하기 위한 가이드입니다. (버전: v0.0.14)

cb-network system의 네트워크 제어 영역(Control plane)을 컨테이너 기반으로 구동하는 방법에 대하여 설명 합니다. 데이터 영역(Data plane)은 pre-built binary를 통해 구동 됩니다.

- Control plane 컴포넌트:
  - cb-network controller,
  - cb-network service,
  - cb-network admin-web
- Data plane 컴포넌트:
  - cb-network agent

비고 - 각 컴포넌트는 독립된 노드(PC, VM, 등)에서 구동 가능합니다. 본 가이드에서는 복잡한 설명을 생략하고자 Control plane 컴포넌트를 한 노드에 구성했습니다.

#### 필요조건(Requirements)

cb-network system의 각 컴포넌트를 설치하기 전에, 아래 사항을 참고하십시오.

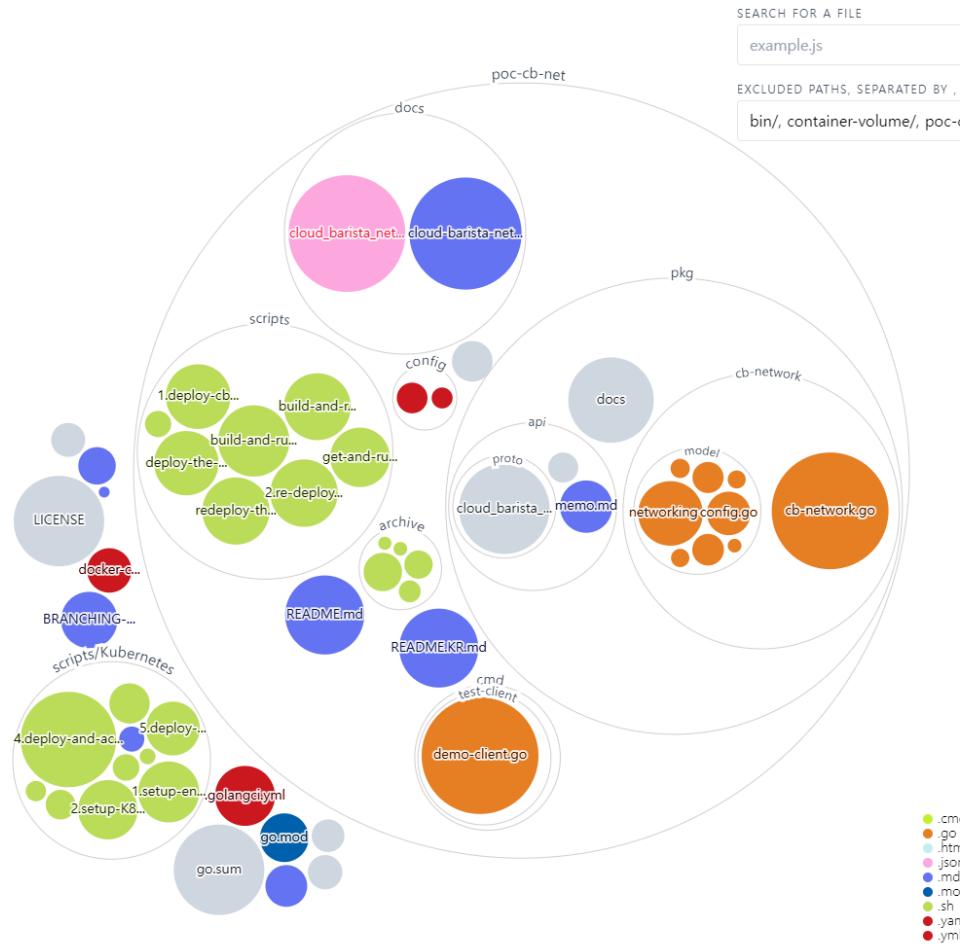
##### 1. VM 3대 준비

- VM 1: Control plane 컴포넌트 설치 및 구동

<https://github.com/cloud-barista/cb-larva/wiki/Install-based-on-container>

# (참고) CB-Larva 디렉토리 계층 구조 및 소스코드

**cloud-barista/cb-larva**



<https://octo-repo-visualization.vercel.app/?repo=cloud-barista%2Fcb-larva>

**\$ tree -d -L 3**

```

.
├── poc-cb-net
│   ├── build
│   └── cmd
│       ├── admin-web
│       ├── agent
│       ├── controller
│       ├── service
│       └── test-client
│   ├── config
│   ├── docs
│   └── pkg
│       ├── api
│       ├── cb-network
│       ├── command-type
│       ├── etcd-key
│       ├── file
│       ├── network-helper
│       ├── network-state
│       ├── rule-type
│       ├── secret-util
│       └── test-type
│   ├── scripts
│   └── web
│       ├── assets
│       ├── public
│       └── scripts
└── Kubernetes

```

golang-standards / project-layout Public Watch 551

Code Issues 51 Pull requests 25 Actions Projects Wiki Security

master 1 branch 0 tags Go to file Add file Code

kcq Merge pull request #121 from flibustenet/patch-1 ... 8f825b7 on 23 Aug 2021 175 commits

Folder	Description	Age
api	Replace broken link in api folder examples.	2 years ago
assets	chore: remove unnecessary .keep files	3 years ago
build	build config file location clarification	3 years ago
cmd	Merge branch 'master' into patch-1	7 months ago
configs	fix a word mistake in writing	2 years ago
deployments	chore: remove unnecessary .keep files	3 years ago
docs	Dapr examples for cmd, docs, pkg and tools	2 years ago
examples	Merge pull request #27 from FantasticFiasco/chore/romo...	3 years ago
githooks	chore: remove unnecessary .keep files	3 years ago
init	chore: remove unnecessary .keep files	3 years ago
internal	internal/pkg example	17 months ago
pkg	typos	10 months ago
scripts	chore: remove unnecessary .keep files	3 years ago
test	chore: remove unnecessary .keep files	3 years ago
third_party	chore: remove unnecessary .keep files	3 years ago
tools	Dapr examples for cmd, docs, pkg and tools	2 years ago
vendor	Dapr examples for cmd, docs, pkg and tools	2 years ago
web	more notes and notes in major directories	4 years ago
website	Capitalize the H in GitHub	2 years ago



# (공개SW) 함께 만들어 갑니다 ^^

정보, 문서, 코드, 리뷰, 의견 하나 하나가 기여이고, 이를 통해 발전하고 있습니다.

cloud-barista / cb-larva Public

Search or jump to... Pull requests Issues Marketplace Explore

Code Issues 12 Pull requests Discussions Actions Projects Wiki Security Insights ...

main 2 branches 5 tags Go to file Add file Code About

hermitkim Merge pull request #146 from hermitkim1/hermitkim1 204 commits 20 hours ago

.github Install golangci-lint by 20 hours ago

poc-cb-net Update template and 20 hours ago

scripts/Kubernetes Set network env varaiable from 20 hours ago

golangciyml Update template and ignore 8 months ago

BRANCHING-AND-RELEASING... Prepare for Cafe Mocha release 3 months ago

CODE\_OF\_CONDUCT.md Create CODE\_OF\_CONDUCT 7 months ago

Dockerfile Rename the cb-network service to controller 12 days ago

LICENSE Create LICENSE 12 months ago

README.KR.md Prepare for Cafe Mocha release 3 months ago

README.md Update README 3 months ago

go.mod Lay the foundations for APIs yesterday

go.sum Lay the foundations for APIs yesterday

README.md

LICENSE Apache 2.0

Read this in other languages: English, 한국어

**CB-Larva: Cloud-Barista Incubator**

Welcome to Cloud-Barista Incubator (for short CB-Larva) 😊

We incubate (research and develop) the new technologies in order to "Contact to the Multi-Cloud". Proof of concept (POC) of new technologies will be performed in this repository. Contributions are always welcome! 🎉

Note that, you can use and share useful information at Cloud-Barista's coffeehouse.

**Challenges in Cloud-Barista**

CB-Larva mainly considers *multi-cloud network technology (cb-network)* for now. The topics below are wide open.

- cb-storage: Multi-cloud storage technology to support storage for the distributed cloud services
- cb-secret: Secret(e.g., configs, credentials, DB access information) management independently and efficiently from source repositories. It will be researched in a private repository (It has secrets!). If you have any interest in it, feel free to contact me.
- but not limited.

We're waiting for your creative ideas. 😊

**정보/문서 공유**

이것은 볼 필요로 중복 기여를 원회하기 위한 대략적인 가이드입니다.

**코드 공유**

1. Issues 템에서 Issue를 생성 (제목을 명화하게 적어주세요.)  
2. Branch를 생성

## 2021 오픈소스 컨트리뷰션 아카데미 Contribution Academy

Cloud Barista's Coffeehouse is an open space for open-minded people who want to share and discuss technical knowledge for a great and happy future.

discussion share communication knowledge information talk insight

Cloud-Barista's Coffeehouse

Explanation of different perspectives lowers the entry barriers for future contributors. 😊

The historian Brian Cowan describes English coffeehouses as "places where people gathered to drink coffee, learn the news of the day, and perhaps to meet with other local residents and discuss matters of mutual concern." (See English coffeehouses in the 17th and 18th centuries)

English Coffeehouse (별칭 Penny House)는 1 Penny 가격의 커피 한잔을 시ما여 지식인들이 다양한 의견을 공유하는 사교 클럽이었습니다.

이자리 Cloud-Barista's coffeehouse에서 다양한 정보를 "편하게" 공유하셨으면 좋겠습니다.

자유롭게 정보, 설명, 의견을 공유하시면서 오픈소스 프로젝트에 참여하시고, 자연스럽게 Cloud-Barista의 여러 Repository에서 기여 포인트를 찾으시며, Contributor, Reviewer, Maintainer로 거듭나실 있으시오! 😊

아래 예시를 포함하여 많은 설명/정보를 기대합니다. 😊

- 클라우드 또는 멀티 클라우드 개념
- Cloud-Barista 관련 용어, 기술, 개념 정리
- Microservice architecture (MSA)
- gRPC (Remote Procedure Call)
- Github Container Registry (GCR)
- Github Actions
- Golang CI/CD (Continuous Integration/Continuous Delivery)
- Cloud-Barista 이슈 및 헬프 빙어
- 설치/배포 자동화 Script

입문자 시작에서 쉬운 설명은 미래 Cloud-Barista Contributor에게 큰 도움이 될 것 입니다. 😊

미래 컨트리뷰터를 위한 메시지 🎉🌟⭐👍

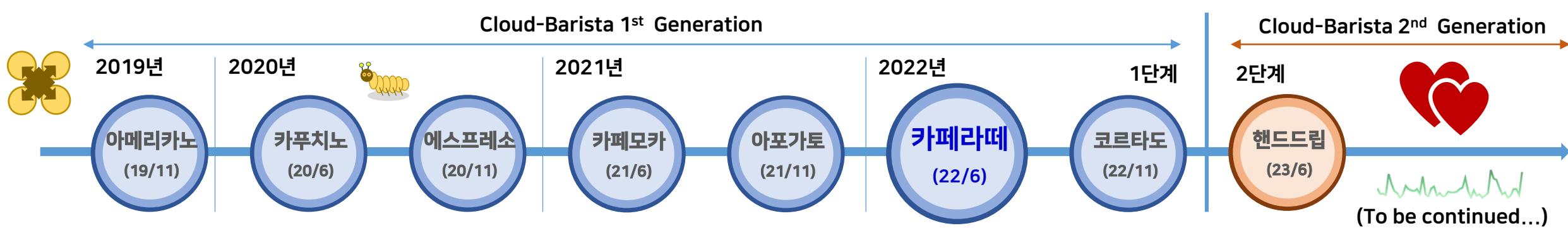
### Learning about the public cloud

#고현경 #남기백 #박범수 #박소연 #박인호 #손현준 #안태건 #유재혁 #이도훈 #이장훈 #임승경 #조성빈 #최수현 #최자현

## Valuable guides to create virtual machine instances on the public cloud

- Amazon Web Services (AWS) - Instance Creation Tip by AWS Go SDK
- Amazon Web Services (AWS) - Creating an AWS EC2 Instance
- Microsoft Azure (MS Azure) - Creating and accessing an instance on MS Azure
- Google Cloud Platform (GCP) - Creating and accessing an instance on GCP
- Alibaba Cloud - Creating and accessing an instance on Alibaba Cloud

# 연구 개발 현황 및 향후 계획



완료    진행    예정

(Agile) 기술 분석, Cloud-Barista Network 시스템 설계 및 업데이트

Cloud Adaptive Network 프로토타입 개발

(외부 연계) cb-network service 제공 및 프레임워크간 연계

(기능 추가) End-to-end 암호화 기능, 성능 평가 기능

(주요 기술 개발) 멀티클라우드 인프라 및 응용을 위한 네트워크

Usecase: 멀티클라우드에 단일 Kubernetes 클러스터 배치 및 운용

(자동화) 개발 워크플로 자동화 적용(CI/CD: Continuous Integration / Continuous Delivery/Deployment)

그 밖에 (cb-loadbalancer, cb-namingservice, 등)

더욱 열심히 채워보겠습니다 ^^;;

악마는 디테일에 있다  
(The devil is in the detail)

# Topics

Multi-cloud network

Multi-cloud storage

Event-driven architecture

Micro-service architecture

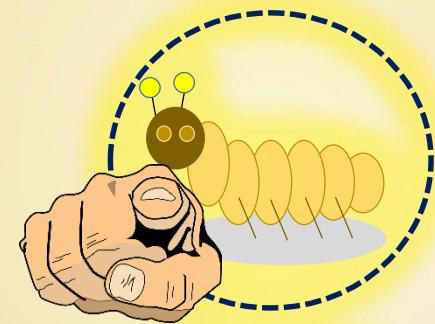
Golang

Web frontend

Distributed key-value store

Distributed storage

# WANTED



CB-Larva

[참고] CB-Larva 저장소: <https://github.com/cloud-barista/cb-larva>

[참고] Cloud-Barista's Coffeehouse 저장소: <https://github.com/cloud-barista/cb-coffeehouse>

[참고] Cloud-Barista Community의 공개SW 활동 비전 및 영상: <https://youtu.be/JOwmFLMxc1w>



## 멀티클라우드 서비스 공통 플랫폼

### 부 록

cb-network 시스템 구조 및 Cloud Adaptive Network 주요 기술 설명

카페라떼(Cafe Latte) 한잔 어떠세요 ?

# (참고) cb-network “컨트롤” 플레인

#성능 #안정성

(참고, 멀티클라우드의 서비스 컨트롤 플레인 배치 모델 5)

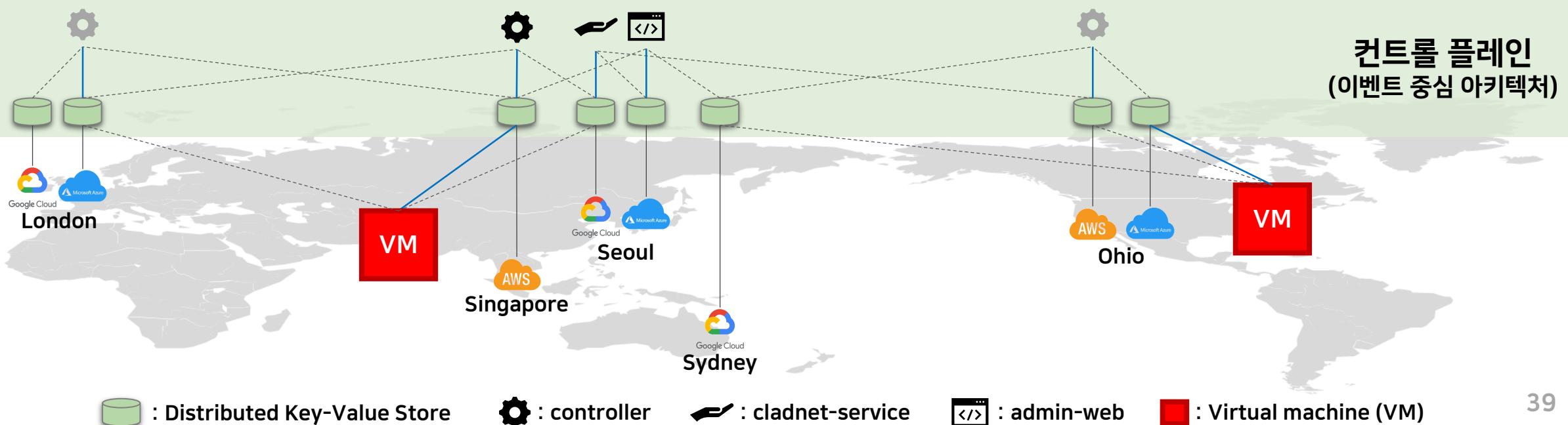
성능 및 안정성을 위한 분산 배치 (by 멀티클라우드의 서비스 컨트롤 플레인 배치 연구)

→ 글로벌 스케일 관점에서 Fault Tolerance(FT) 및 지역적(Region) 성능 품질을 고려한 배치

예1 - 지역적 성능) Cloud A의 Seoul region이 다운 되어도 클라우드 B의 Seoul region에서 데이터 이용 가능

예2 - 장애 허용) 천재지변으로 인해 Region A의 모든 CSP에 장애가 발생시 Region B에서 데이터 이용 가능

분산 배치를 통해: ✓ (거리/속도) 네트워크 운용 성능 최대화      ✓ (안정성) 결함 허용을 위한 네트워크 정보 다중화

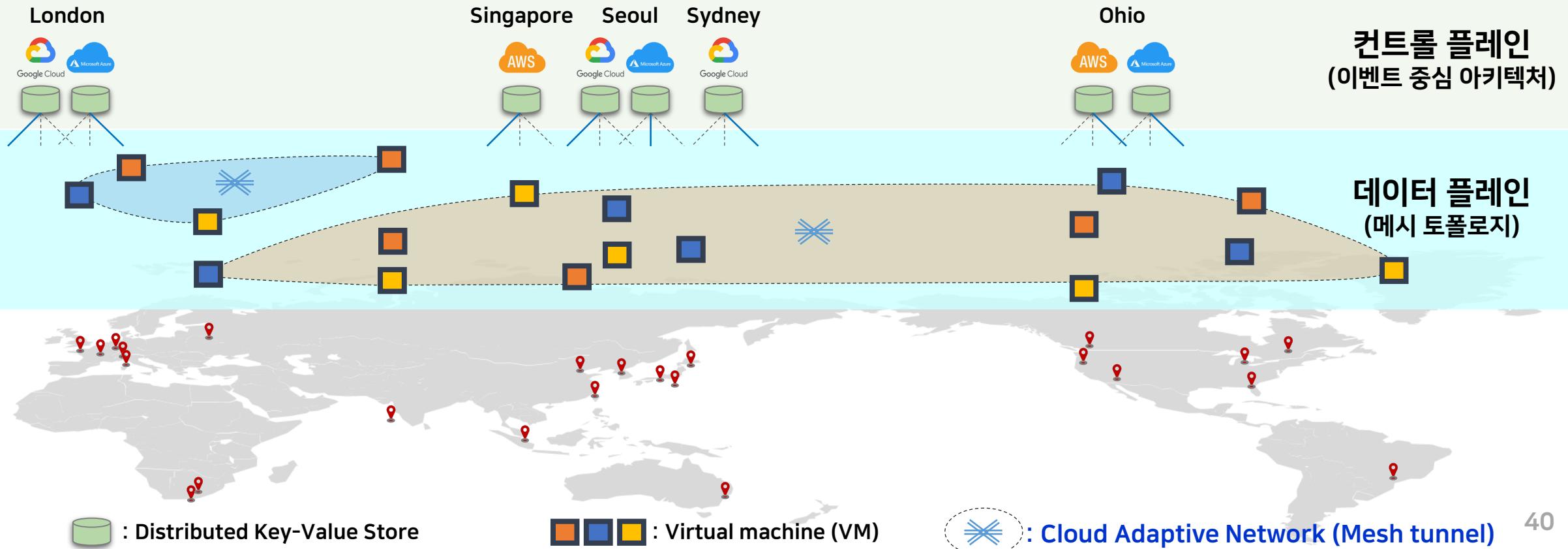


# (참고) cb-network “데이터” 플레인

#성능 #가변성

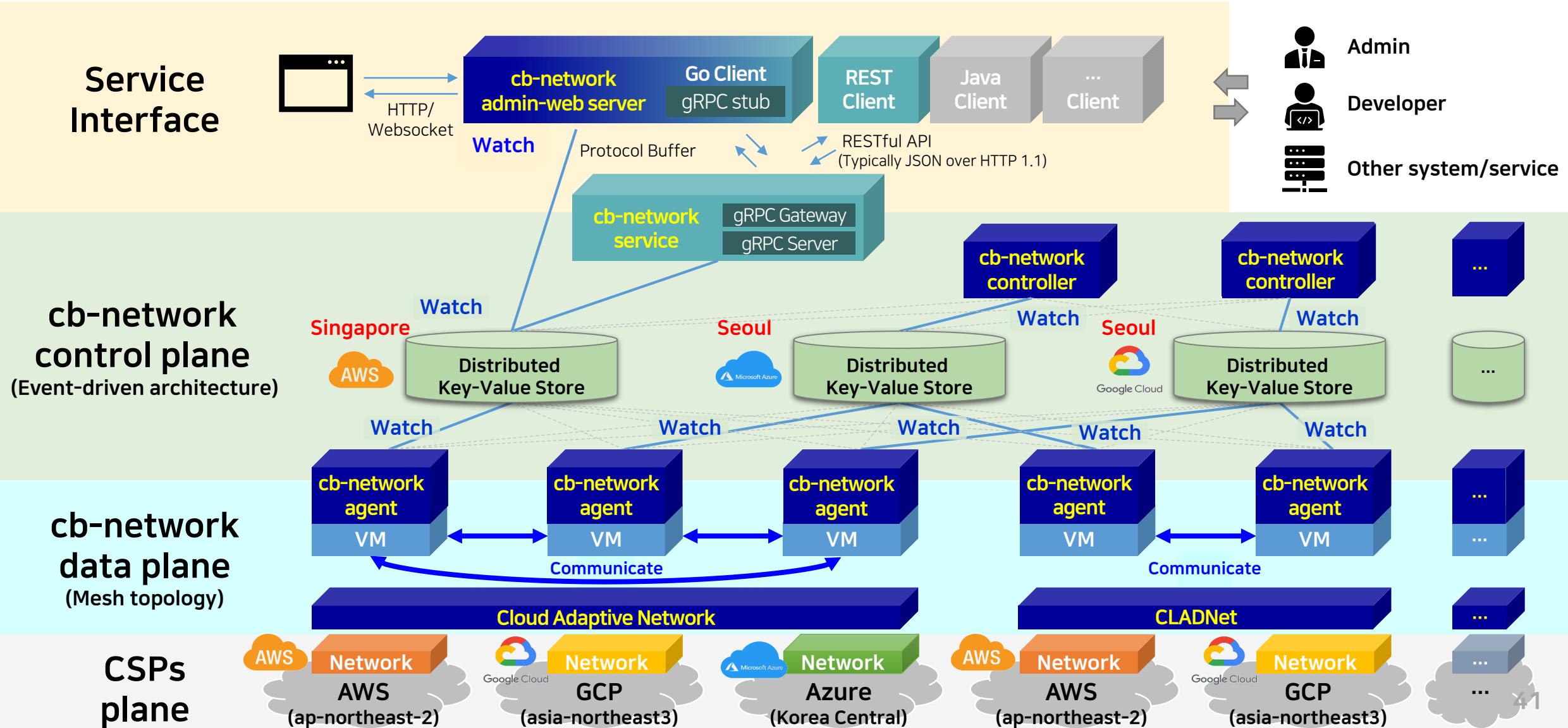
이벤트 중심 처리: 네트워크 상태 동적 업데이트 / 메시 토플로지: 통신 속도 지원

컨트롤 플레인을 통해: ✓ (가변성) 변경되는 네트워크 정보 동적 동기화 ✓ (거리/속도) 메시 토플로지를 기반으로 통신 속도 지원 (증개노드 X)



# cb-network 시스템의 세부 구조 및 컴포넌트

#구조 #복잡 #POC수행중



# Cloud Adaptive Network

Cloud Adaptive Network (CLADNet): 멀티클라우드 가상/공통 네트워크 기술

clad: 차려입은, 장비한

- ✓ 서로 다른 클라우드의 서로 다른 서브넷 상에서 인프라 및 응용(VM, Container 등)들이 동일 서브넷에 존재하는 것처럼 사실 IP 기반으로 운용, 관리할 수 있도록 하는 기술
- ✓ 멀티클라우드의 다양한 네트워크에 적응가능한 오버레이 네트워크로 VM 그룹에 동일 네트워크를 제공함



상이함(다양성)과 가변성을 해결하는 기술 ^^



## Check points

---

상이함과 가변성 해결을 위한 Check points:

✓ 유동성

✓ 중복성

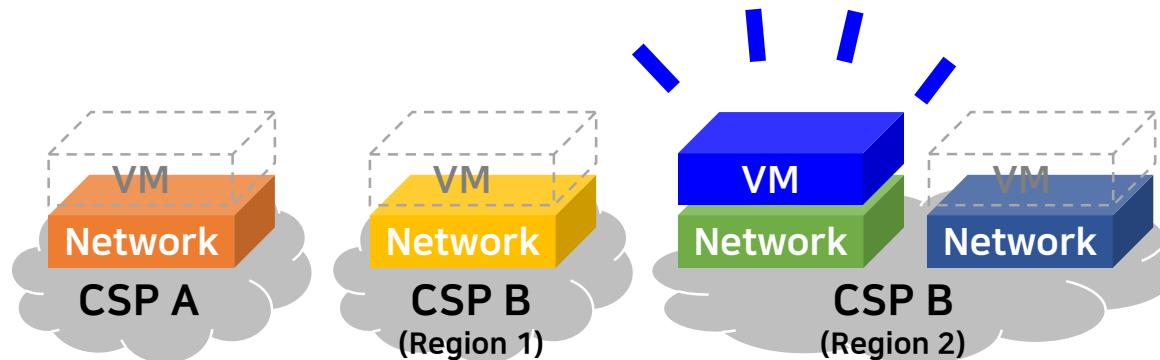
✓ 변동성

✓ 확장성

## 체크 포인트 - 유동성

클라우드 자원은 유동적이다!?

- 인프라(MCIS) 생성 전에는 정확한 네트워크 정보를 얻기 어려워요.

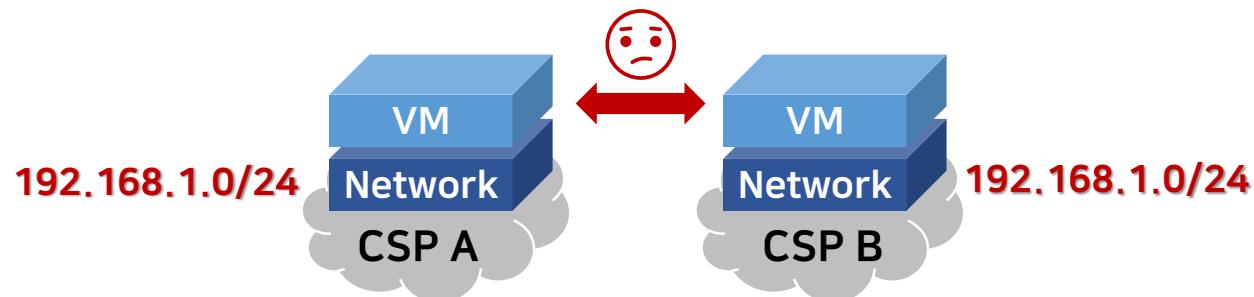


## 체크 포인트 - 중복성

클라우드간에 중복된 Subnet이 생성 될 수 있다?!

→ VPN Gateway를 활용해도 통신이 어려울 수 있고, Supernetting\* 적용 또한 어려워요.

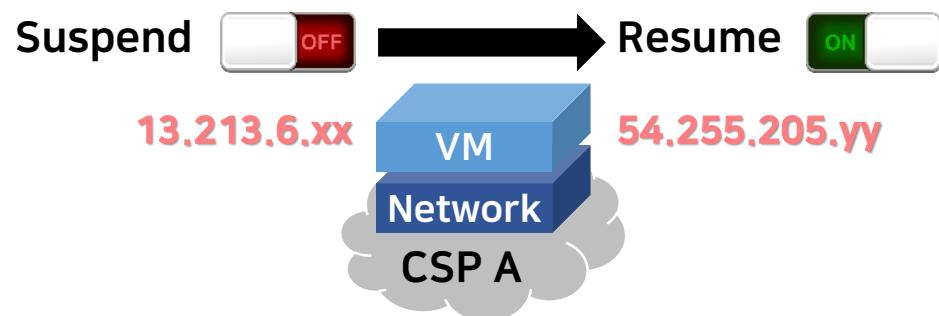
\* 여러 주소 영역을 하나의 주소 영역으로 통합하는 개념 (Subnetting과 반대되는 개념)



## 체크 포인트 - 변동성

VM관련 네트워크는 변경될 수 있다?!

- 통신에 문제가 발생 할 수 있어요. (i.e., Suspend → Resume시 아이피 주소 변경)



## 체크 포인트 - 확장성

글로벌 스케일 인프라?!

- 적합한 규모의 IPv4 private address space를 제공하지 못할 수 있어요.



체크포인트 해결을 위한

## Cloud Adaptive Network 주요 기술

가상/공통 네트워크 제공

네트워크 정보 동적 업데이트

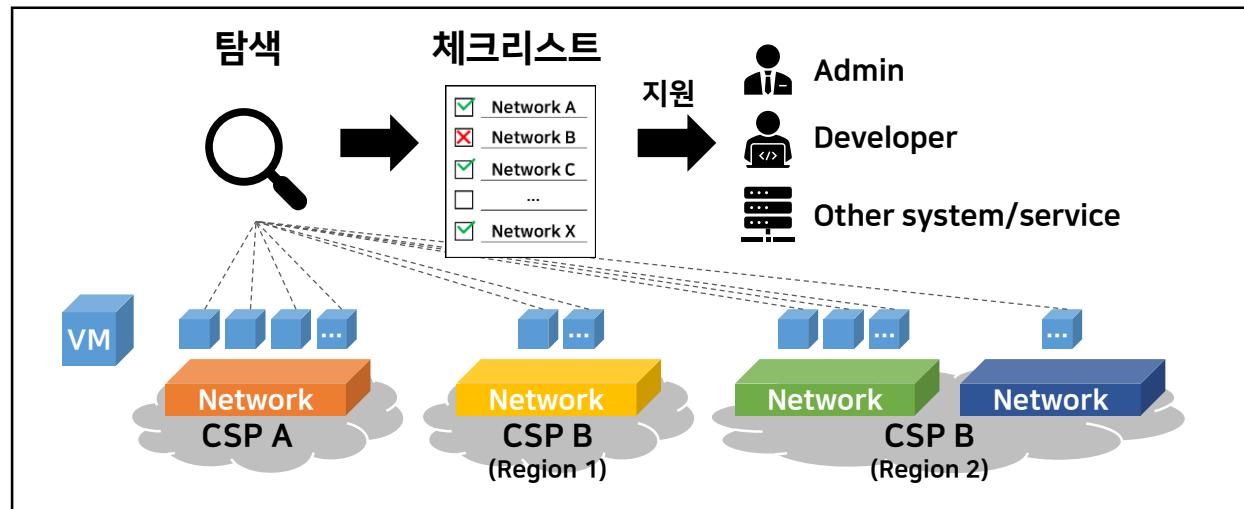
네트워크 터널링(Tunneling)

비용 우선적 네트워킹

# Cloud Adaptive Network 주요 기술 1

## 가상/공통 네트워크 제공:

가용한 IPv4 사설 주소 공간(private address spaces) 분석 및 추천 서비스 제공

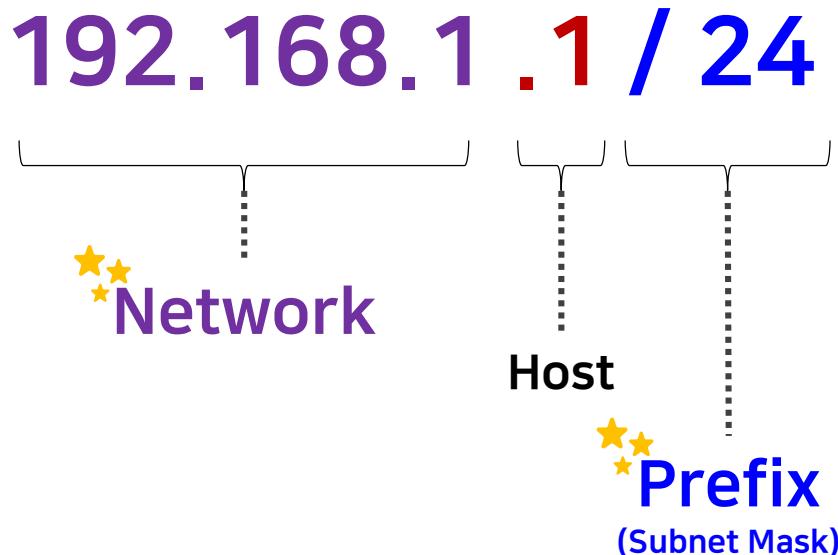


### → 가용한 주소공간 리스트 제공

- 10.0.0.0/8 이하 가용한 대역
- 172.16.0.0/12 이하 가용한 대역
- 192.168.0.0/16 이하 가용한 대역

### → 대상 노드(VM) 수를 바탕으로 적절한 규모의 주소공간 추천

# 가상/공통 네트워크 제공 기술

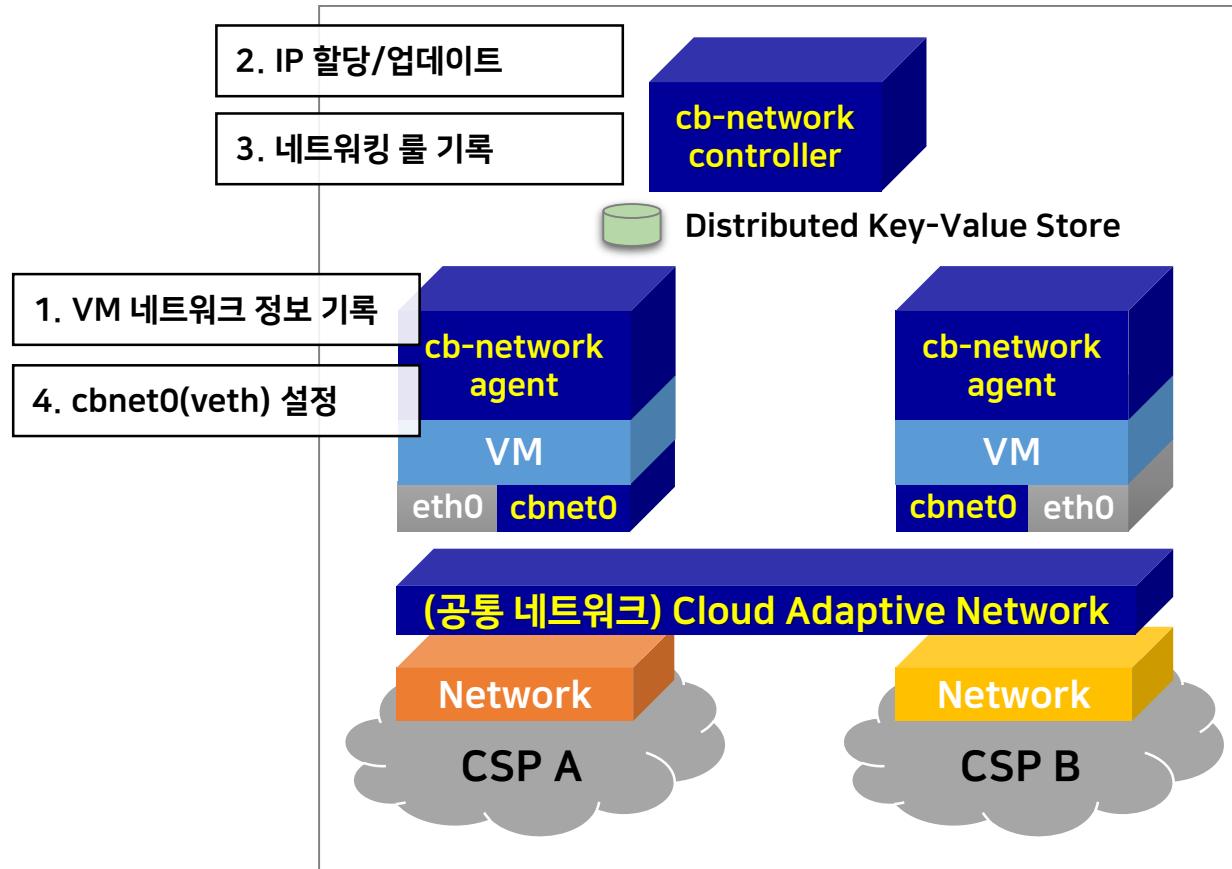


- ✓ CIDR(Classless Interdomain Routing)에서 접두사(prefix)는 네트워크를 정의하는 데 사용됨
- ✓ 따라서, 사용하지 않는 접두사를 기반으로 사용 가능한 IPv4 사설 네트워크 공간을 찾을 수 있음

# Cloud Adaptive Network 주요 기술 2

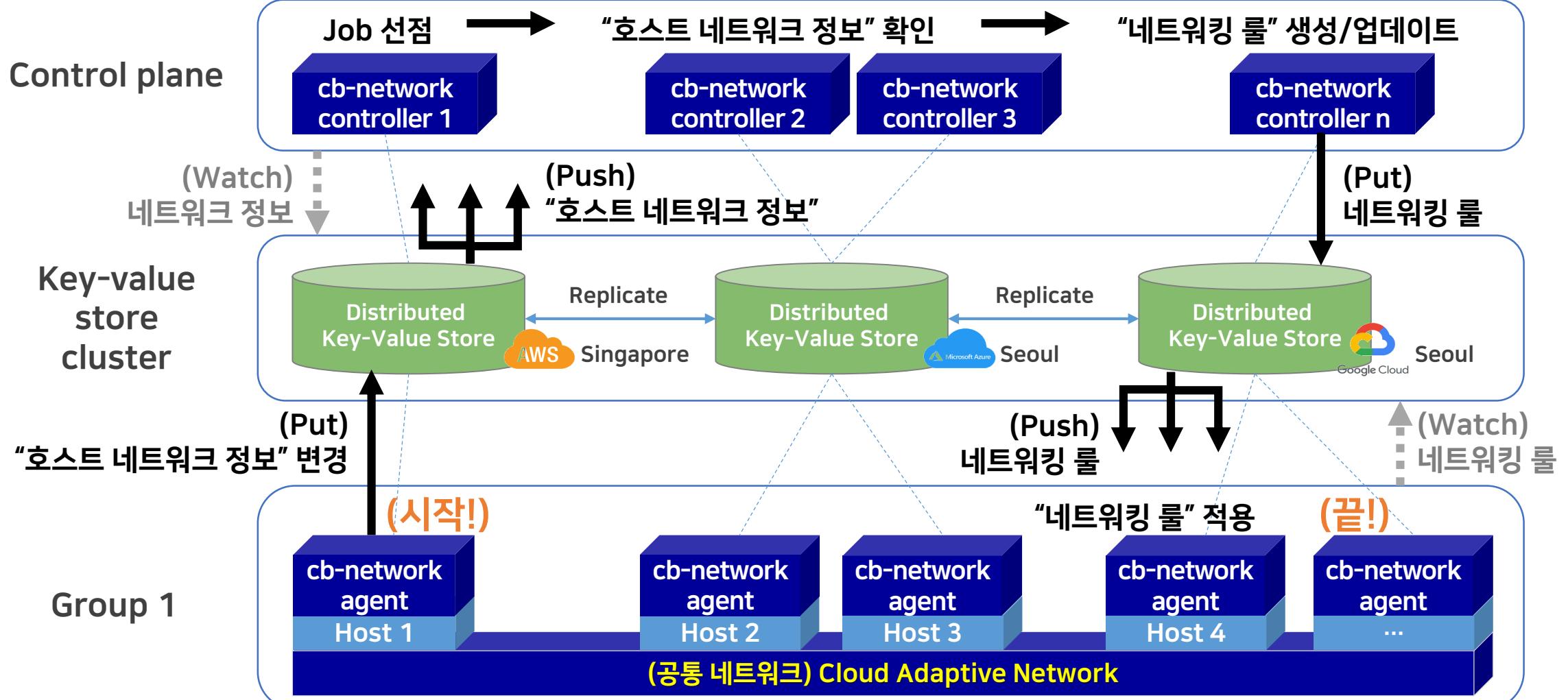
## 네트워크 정보 동적 업데이트:

VM 네트워크 정보 동적 업데이트, 네트워킹 룰 사전 공유 및 동기화



- Event-driven 아키텍처 채택으로 네트워크 변화 동적 업데이트 (by Distributed key-value store (i.e., etcd)의 Watch)
- VM 네트워크 변화에 따라 네트워킹 룰 동적 업데이트 및 동기화
- VM 가상/공통 네트워크 인터페이스 (cbnet0) 할당

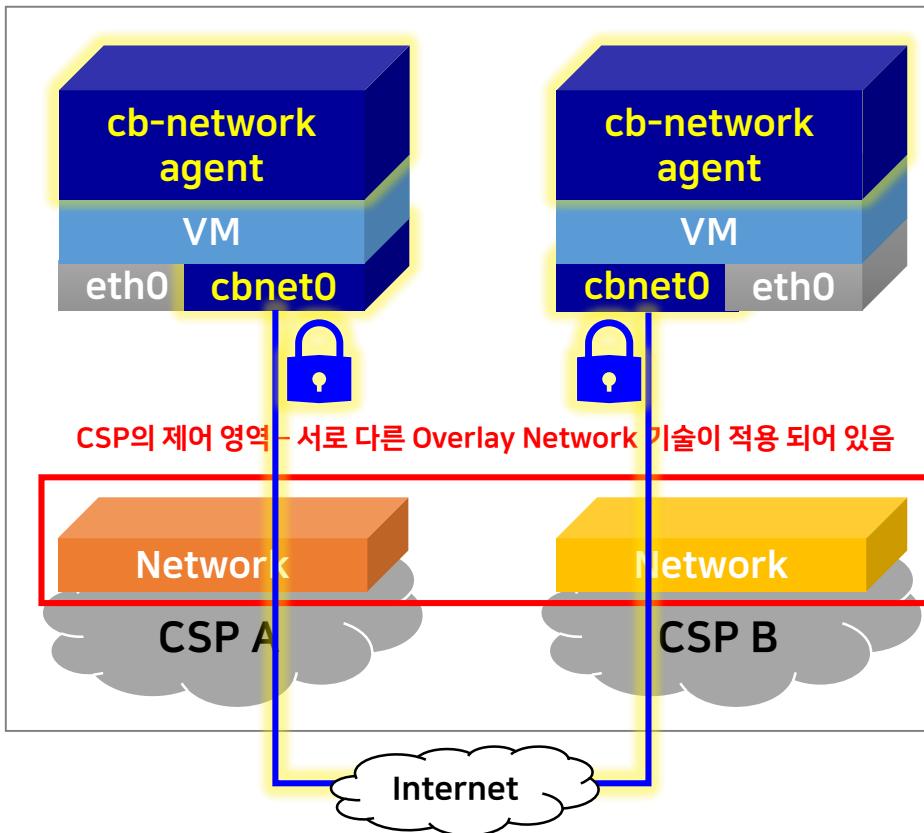
# 네트워크 정보 동적 업데이트 기술



# Cloud Adaptive Network 주요 기술 3

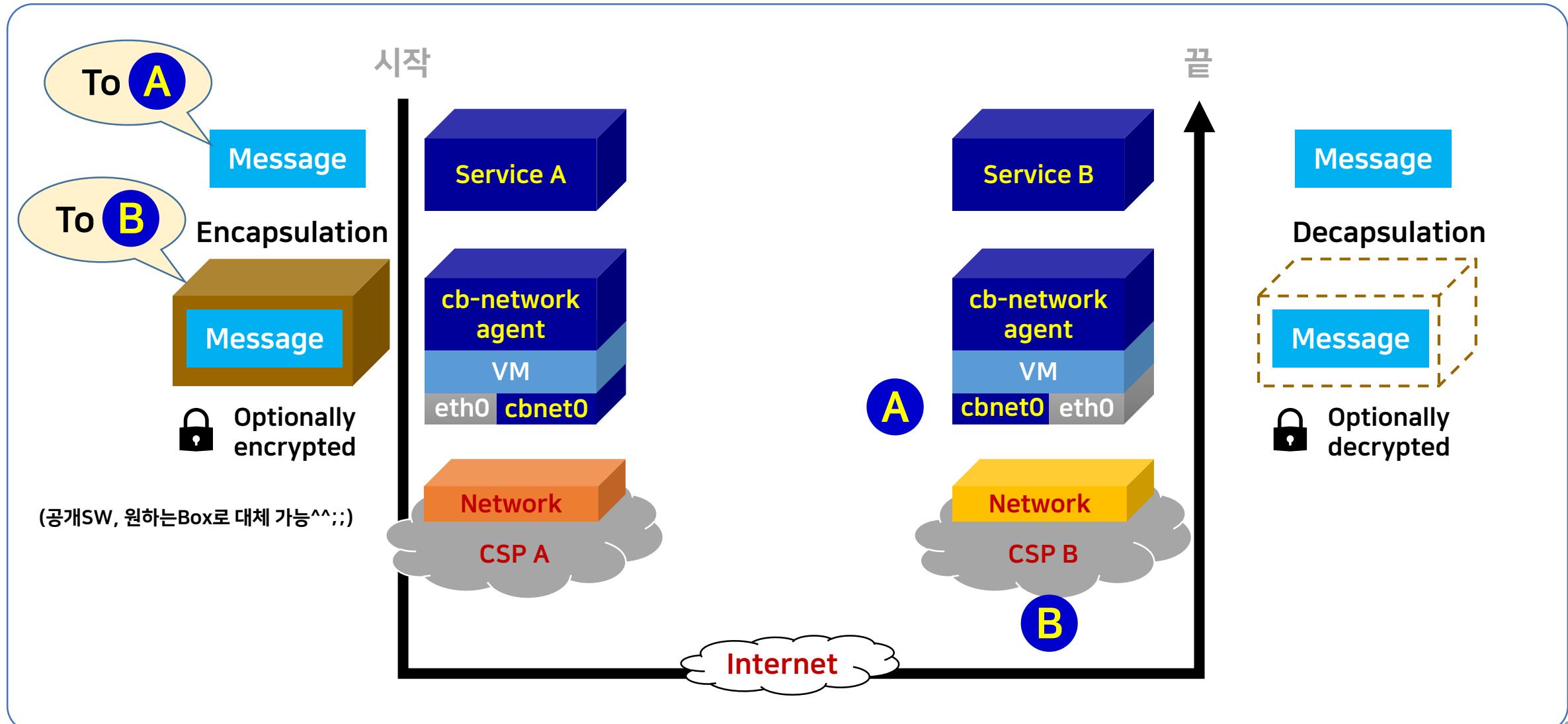
## 네트워크 터널링(Tunneling):

Tunneling for CSP의 제어 영역을 “통과” 하는 통신 (+ End-to-end encryption (optional))



- 서로 다른 Overlay Network 기술이 적용된 CSP의 제어 영역을 통과하여 통신 가능
- 선택적으로 End-to-end encryption 적용 가능

# 네트워크 터널링 기술



# (참고) 연구개발용 UI

#cb-network admin-web  
#websocket 적용

## Cloud Adaptive Network 생성 및 확인 (CSP의 웹 콘솔/포털 벤치마킹)



**Cloud-Barista**

The AdminWeb of Cloud-Barista Network

See AdminWeb

**CLADNet: Cloud Adaptive Network for Multi-cloud**

Cloud Adaptive Network list

ID	Name	IPv4 CIDR block	Description
c8e7kdbn5bcfkiregbg	cladnet01	192.168.0.0/26	test cladnet
c8ffu3rn5bc06nth0og	cladnet02	172.16.0.0/24	test cladnet

Create your Cloud Adaptive Network

Name:

Network (IPv4 CIDR block):  (IP address / Subnet mask will be converted and provided)

Description:

**Create**

## Cloud Adaptive Network 구성 및 제어 (Multi-CLADNet 지원)

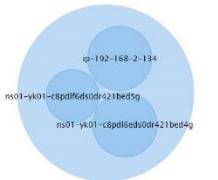
**Cloud Adaptive Network configuration**

The Cloud Adaptive Networks

Remote-control

CLADNet ID: Choose here

Resume Suspend



CLADNet c8pdoh2tih9mg62490

Highcharts.com

**Networking rule**

Host ID	Host IP CIDR Block	Host IP Address	Public IP	State
ip-192-168-2-134	192.168.0.2/28	192.168.0.2	52.77.226.145	running
ns01-yk01-c8pdif6dsodr421bed5g	192.168.0.3/28	192.168.0.3	128.0.44.26	running
ns01-yk01-c8pdif6dsodr421bed4g	192.168.0.4/28	192.168.0.4	34.81.33.131	running

## Cloud Adaptive Network 상태 확인 (모니터링 기능에 활용 가능)

The status Cloud Adaptive Network: connectivity and latency

CLADNet ID: c8pdoh2tih9mg62490

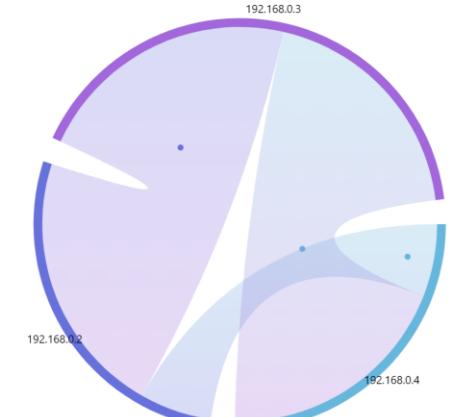
TrialCount: 3

Execute

**Status table**

Source	Destination	Latency (ms)
192.168.0.4 (ns01-yk01-c8pdif6dsodr421bed5g)	192.168.0.2 (ip-192-168-2-134)	46.58
192.168.0.4 (ns01-yk01-c8pdif6dsodr421bed5g)	192.168.0.3 (ns01-yk01-c8pdif6dsodr421bed4g)	153.42
192.168.0.2 (ip-192-168-2-134)	192.168.0.3 (ns01-yk01-c8pdif6dsodr421bed4g)	170.77

**Status chart**



클라우드 바리스타들의 여섯번째 이야기

# “오픈소스”로 만들어가는 “멀티클라우드” 생태계

Cloud-Barista Community the 6th Conference

# 감사합니다.

<https://github.com/cloud-barista>  
<https://cloud-barista.github.io>

(김 윤 곤 / [contact-to-cloud-barista@googlegroups.com](mailto:contact-to-cloud-barista@googlegroups.com))