



# 멀티클라우드, “새로운 생태계를 향한 클라우드 비긴어게인”

## 클라우드바리스타 커뮤니티 제5차 컨퍼런스

[세션] CB-Larva:

## 멀티클라우드 가상 네트워크

김 윤 곤

CB-Larva 인큐베이터 리더

아포가토(Affogato) 한잔 어떠세요 ?

# 이번 세션은?

응용/도메인/기관 특화 SW



멀티클라우드 서비스 개방형 인터페이스

멀티클라우드 애플리케이션  
통합관리 프레임워크

멀티클라우드 인프라 서비스  
통합 관리 프레임워크

멀티클라우드 인프라 연동  
프레임워크

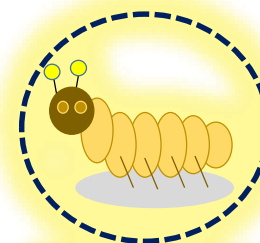
멀티클라우드 통합  
모니터링 프레임워크

멀티클라우드 서비스 공통 플랫폼

## CB-Larva: Cloud-Barista Incubator



멀티클라우드 기술 실험실



CB-Larva

Cloud-Barista의 정체성을 잃지않고,  
**지속적으로 신규 니즈를 수용**하기 위하여  
**신기술, 부족기술 등의 POC**를 수행하며,  
이를 Cloud-Barista로 흡수하기 위한 기술 인큐베이터

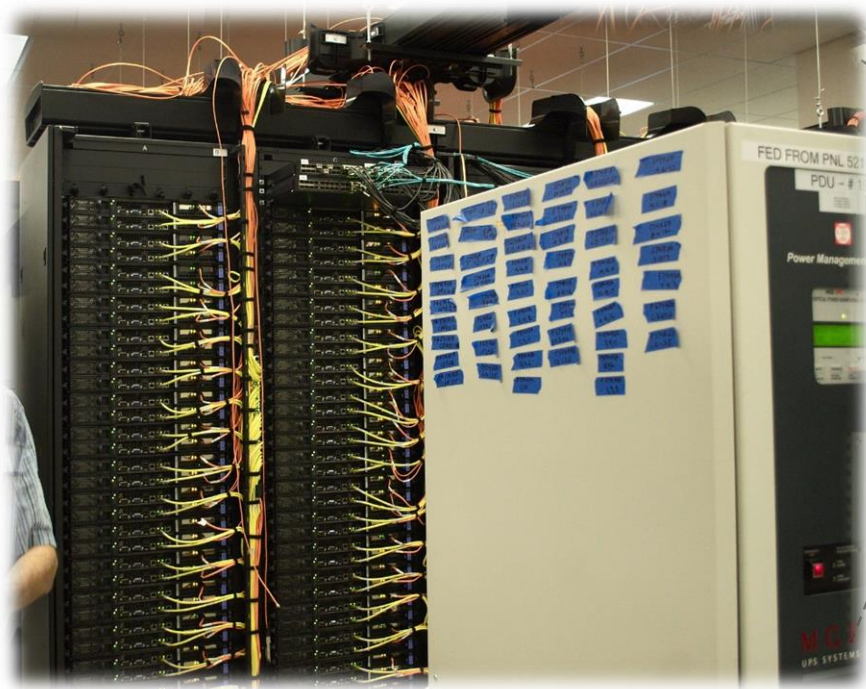
# 이번 발표는 ?

서로 다르고 ...

변하고 ...



데이터 센터의 “네트워크”는?



멀티클라우드의 “네트워크”는?

시스템

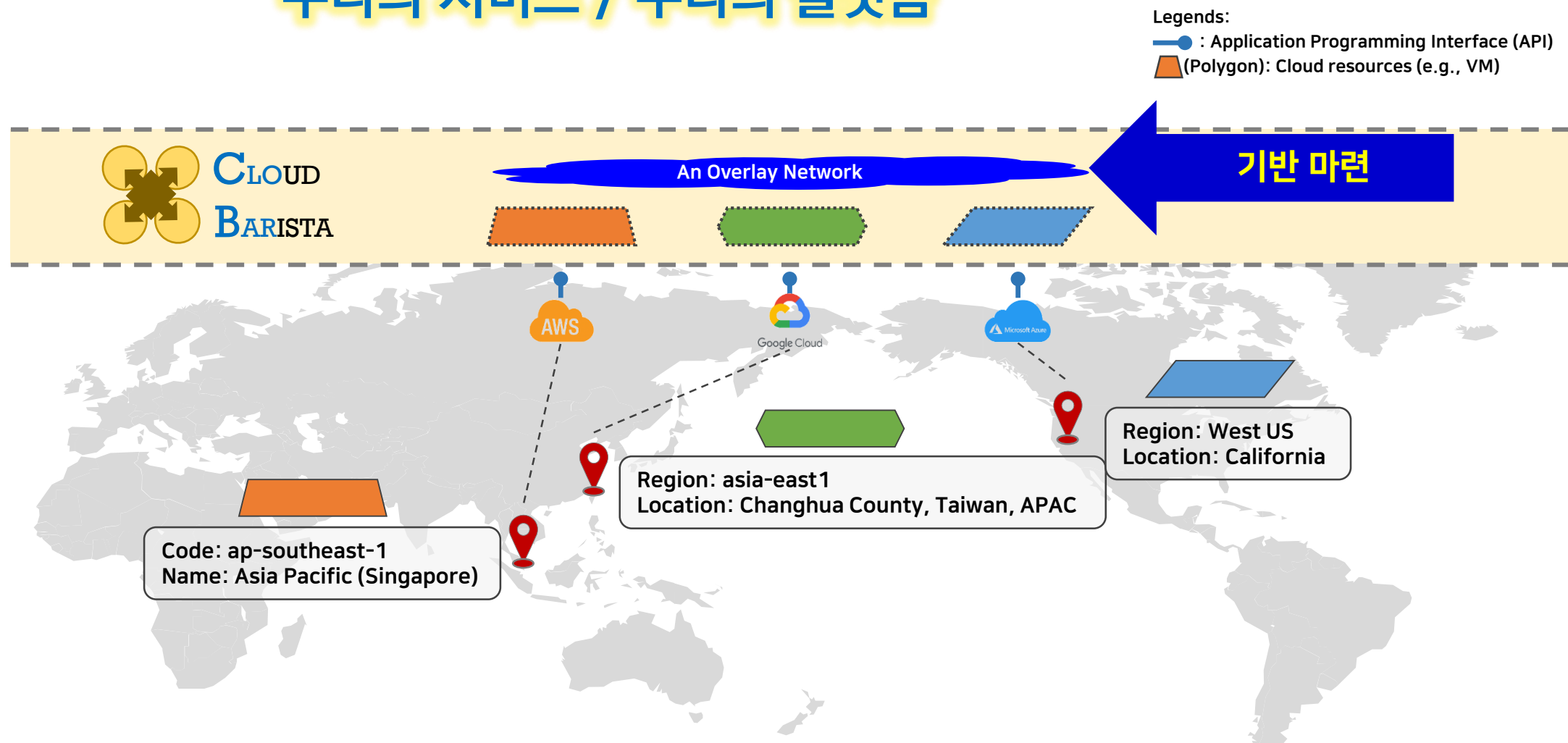
기술

오늘 많이 등장할  
세계지도 ^^;;



# 어디에 활용하나요?!

## 우리의 서비스 / 우리의 플랫폼



# Contents

---

**I** 글로벌 스케일 네트워킹

**II** Cloud Adaptive Network 구조 및 주요 기능

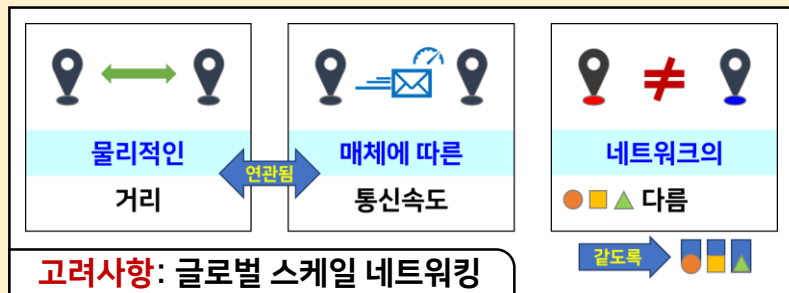
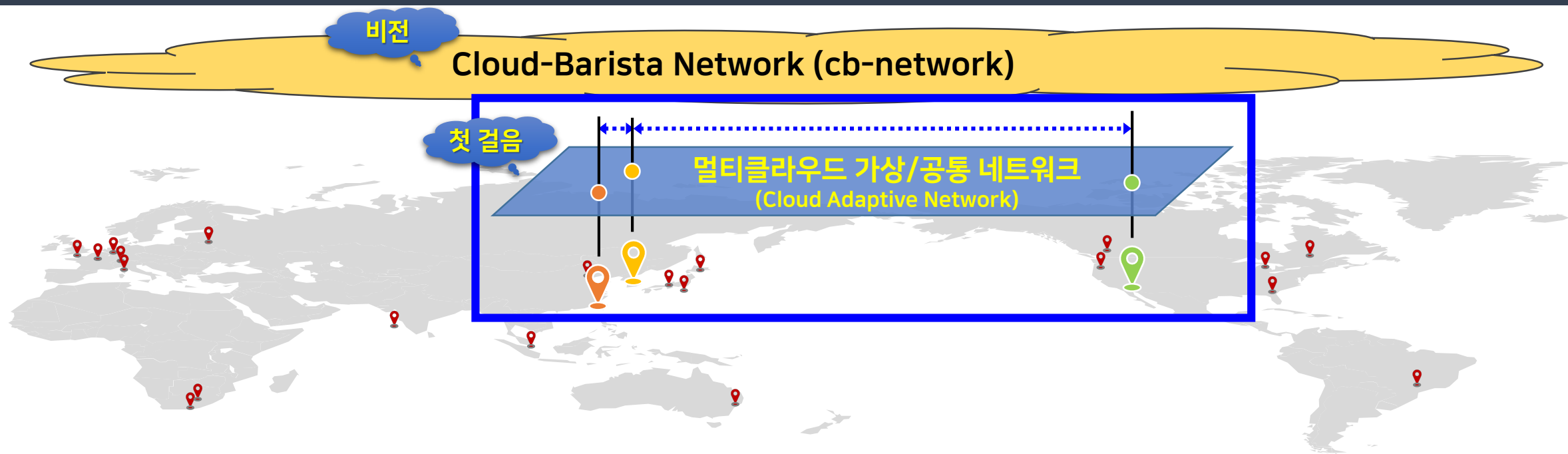
**III** Cloud Adaptive Network 활용 예시

**IV** CB-Larva의 현재, 비전, 그리고 공개SW

# 글로벌 스케일 네트워킹이란?!

\* 멀티클라우드 인프라 및 응용의 통신을 위해 필수!!

전세계 클라우드 인프라를 엮기 위해 여러 클라우드 네트워크의 상이함과 가변성을 해결하는 효율적인 통신 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

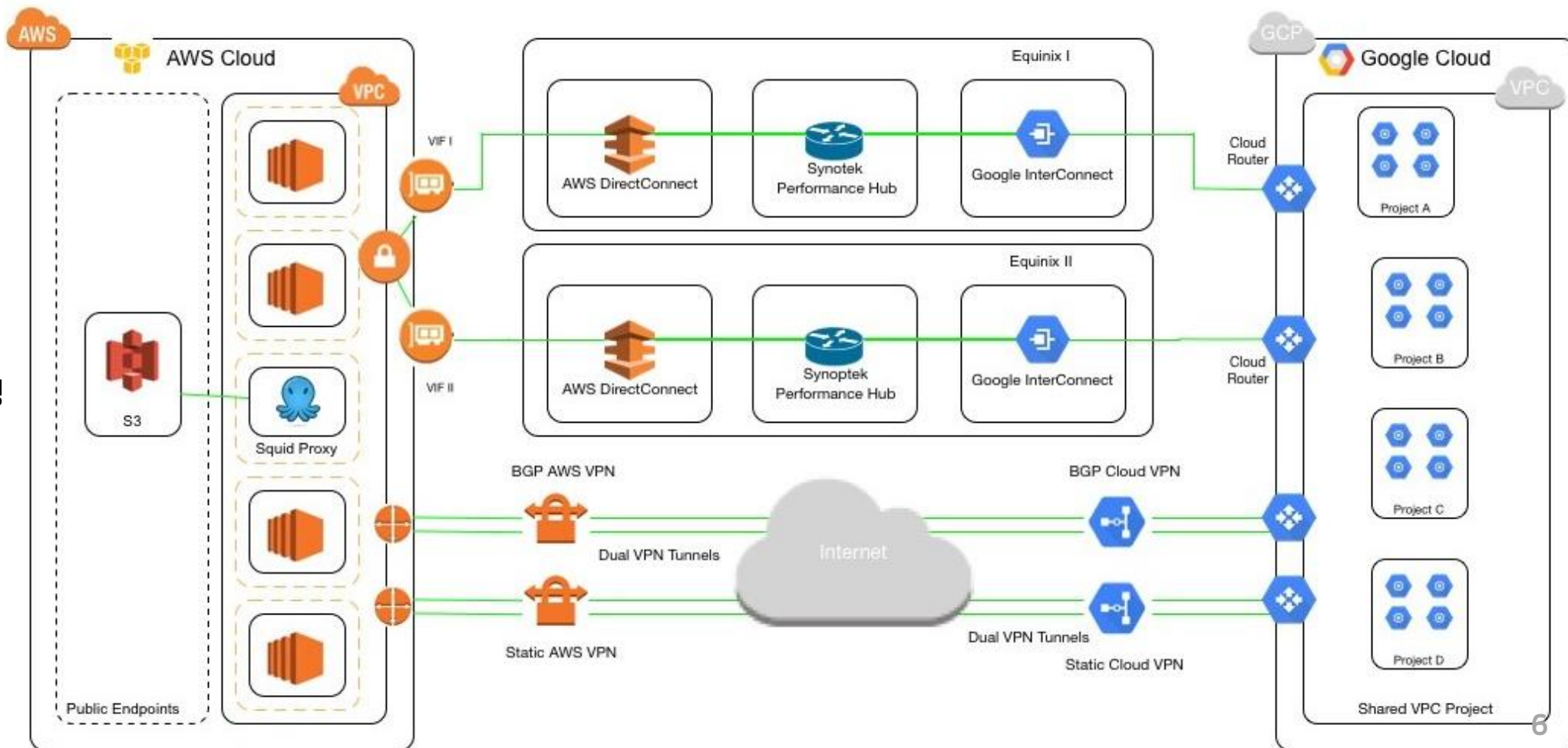


# (현재는...) CSP의 Site-to-Site 네트워킹 서비스

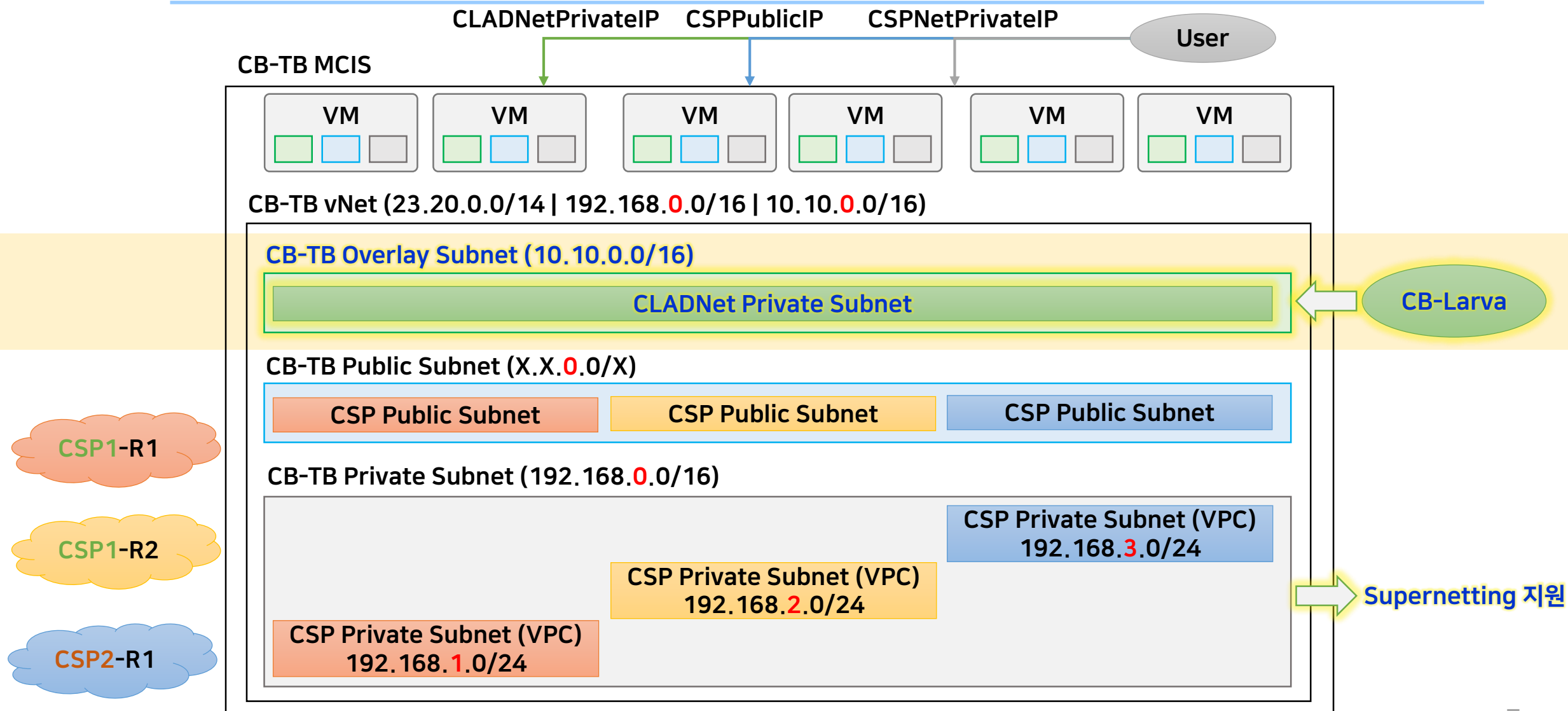
사례: 다른 네트워크와 연결성을 제공하기 위한 CSP의 자체 네트워크 기능(서비스)

## [CSP 서비스]

- 전용선 (및 허브)
  - 정적 CIDR 라우팅
  - BGP 자동 라우팅
- Transit Gateway (클라우드 라우터)

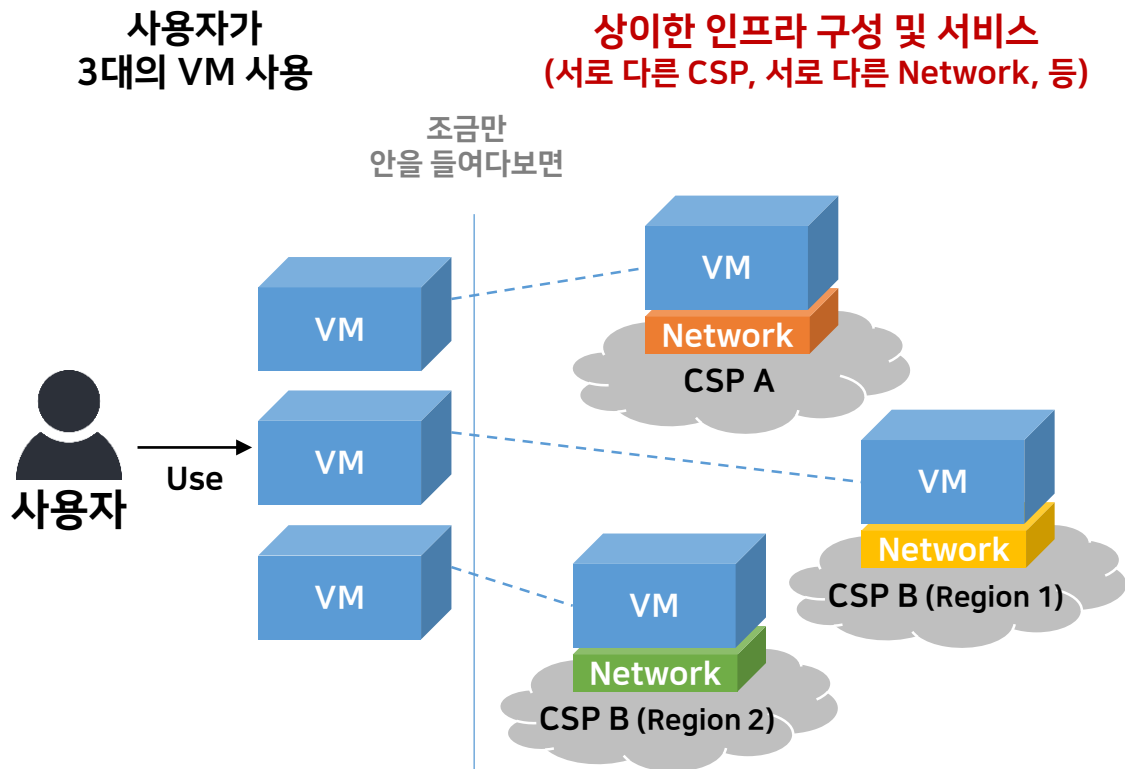


# (우리는...) MCIS 네트워크 구성 형상





# (Challenges) 멀티클라우드를 활용하다 보면



## Challenges:

- ✓ VM은 자신의 Public IP 주소를 몰라요.
- ✓ VM을 suspend → resume하면 Public IP가 바뀌어요.
- ✓ 서브넷이 다르면 K8s/OpenStack등을 설치가 어려워요.
- ✓ CSP의 네트워크를 제어하기 어려워요.
- ✓ 지역적으로 멀~리, 많~이 얹여있다 보니 성능 저하와 오류 발생이 많아요.



연구 개발 중:

- ✓ Cloud-Barista Network “시스템”
- ✓ Cloud Adaptive Network “기술”



## (지향점) Cloud-Barista Network (cb-network)

Cloud-Barista가 바라보는 **글로벌 스케일 네트워크**

- ✓ **Adaptive**: 여러 사업자의 상이한 Cloud Network에 적응 가능한 네트워크  
    ➡ **Cloud Adaptive Network**: 서로 다른 CSP 네트워크에 적응 가능한 네트워크
- ✓ **Fault tolerant**: 사업자 및 리전 이슈에 대비하는 글로벌 장애 허용 네트워크
- ✓ **Lightweight**: 호스트(VM) 자원 사용을 최소화하는 경량한 네트워크
- ✓ **Handy**: 사용자가 쉽고 빠르게 사용할 수 있는 네트워크

# Cloud Adaptive Network (CLADNet)

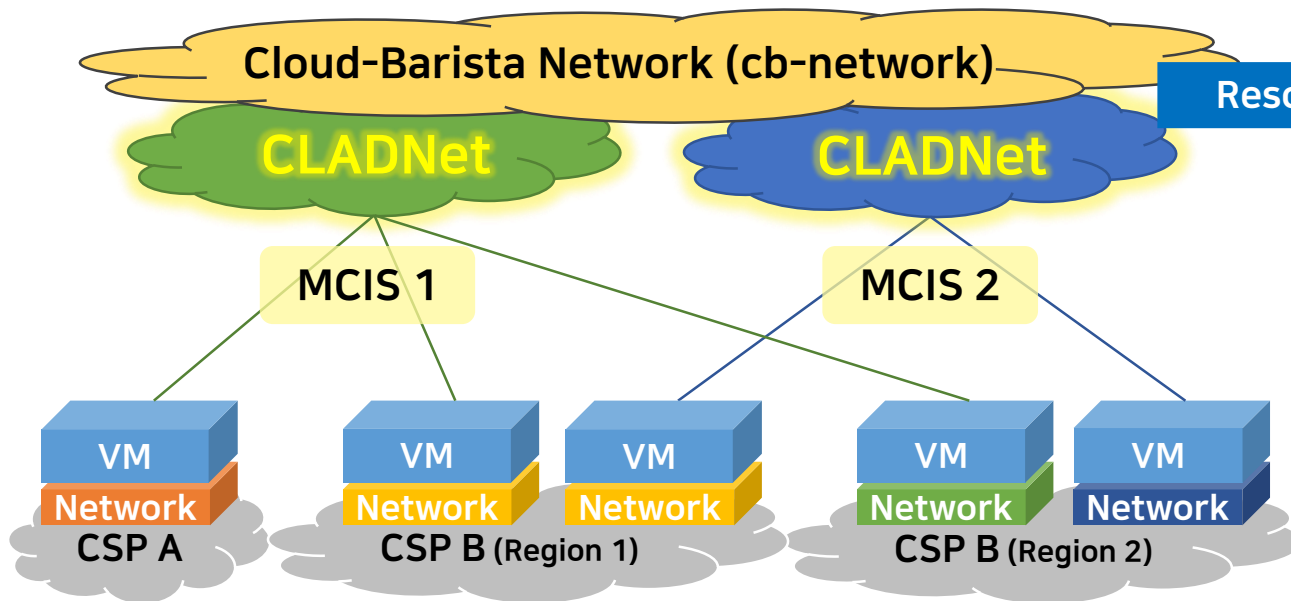
\* clad: 차려입은, 장비한

전세계 클라우드 인프라를 엮는 **글로벌 스케일 네트워킹 기술을 위한 첫 걸음**

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

- 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 논리적인 **노드 그룹**에 동일 네트워크를 제공함

예: Multi-Cloud Infra Service (MCIS)



## Challenges:

- ✓ VM은 자신의 Public IP 주소를 몰라요.
- ✓ VM을 suspend → resume하면 Public IP가 바뀌어요.
- ✓ 서브넷이 다르면 K8s/OpenStack등이 설치가 어려워요.
- ✓ CSP의 네트워크를 제어하기 어려워요.
- ✓ 지역적으로 멀~리, 많~이 엮여있다 보니 성능 저하와 오류 발생이 많아요.

# (구조) Cloud-Barista Network (cb-network)

## 글로벌 스케일 네트워킹을 위한 시스템 구조

### ✓ Event-driven 아키텍처:

분산 Key-Value store 기반의 Event-driven 아키텍처로 데이터의 변경, 생성, 삭제(CUD)시 발생하는 서비스의 의미있는 변화를 바탕으로 효율적인 Workflow 수행

→ Microservice Architecture를 향해 나아가는 중

### ✓ 메쉬(Mesh)형 토폴로지:

오버레이 네트워크를 메쉬형 토폴로지로 구성하여 중개 노드의 위치에 따른 성능 차이를 최소화함

➡ 멀티클라우드 네트워크의 성능, 안정성, 가변성 지원을 위해 꼭 필요한 구조

# (성능/안정성) cb-network “컨트롤” 플레인

(참고, 멀티클라우드의 서비스 컨트롤 플레인 배치 모델 5)

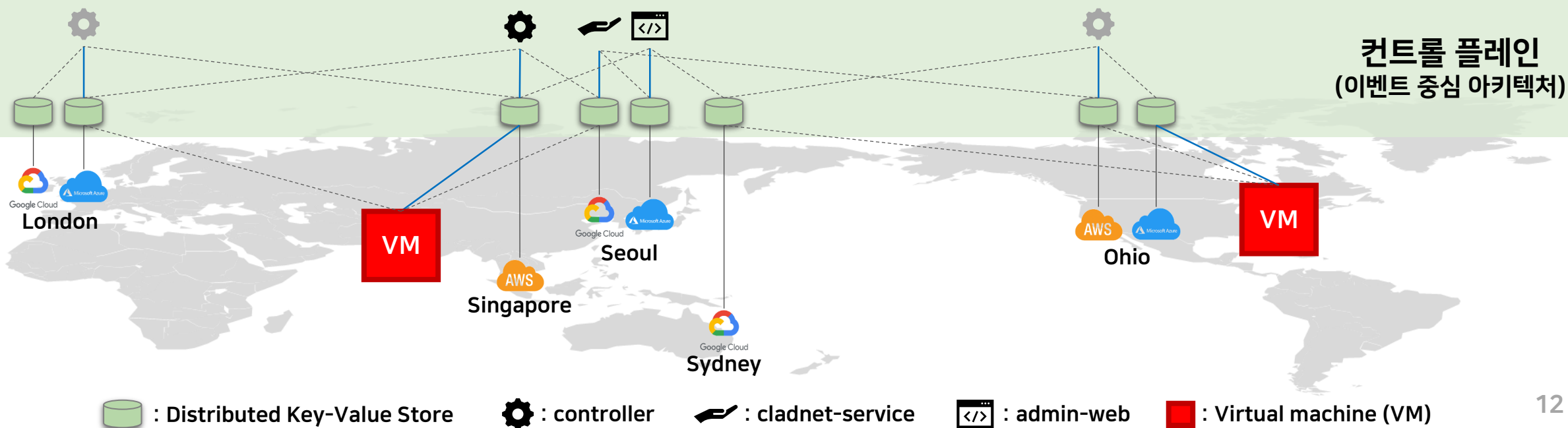
최적의 cb-network 시스템을 구축하려면...? 컨트롤 플레인의 Hosts를 여러 CSP의 여러 Region에 배치 ?!

➡ 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

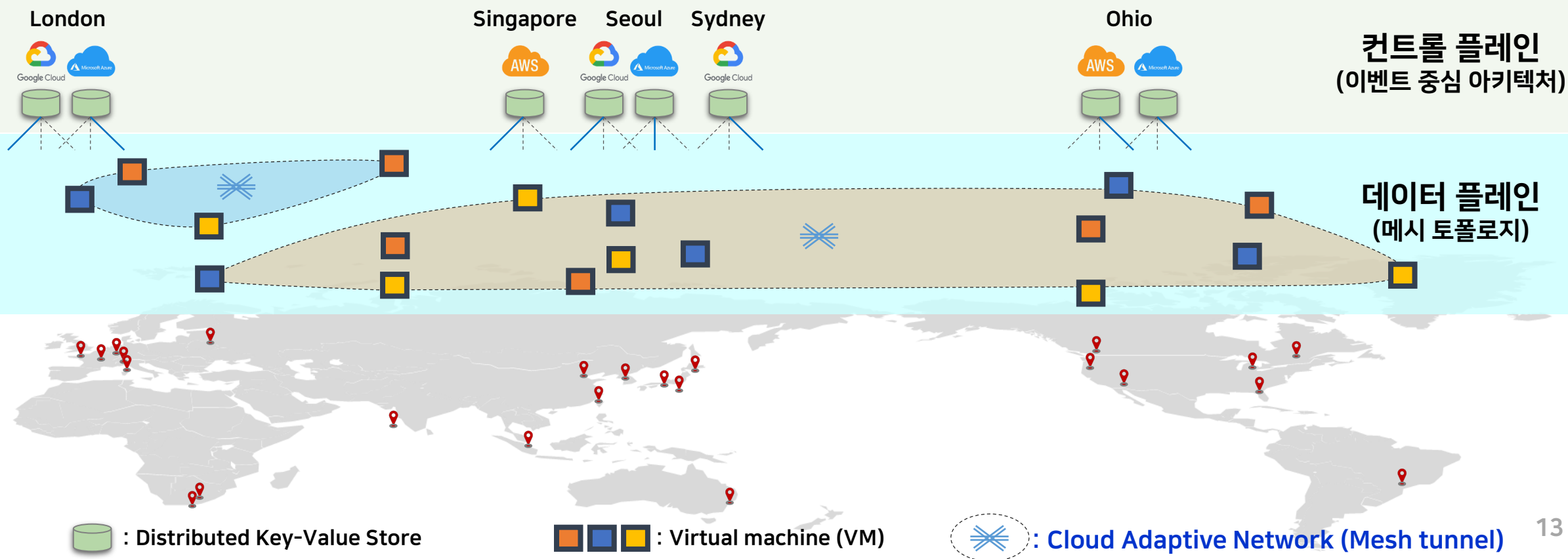
분산 배치를 통해:    ✓ (거리/속도) 네트워크 운용 성능 최대화    ✓ (안정성) 결함 허용을 위한 네트워크 정보 다중화



# (성능/가변성) cb-network “데이터” 플레인

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

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







복잡 하지만... POC 수행중인 구조 입니다.

# (전체구조) 서비스 제공을 위하여

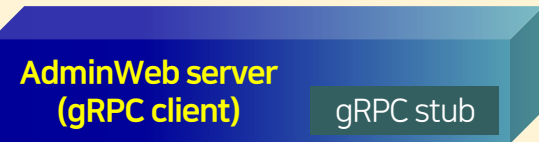
#gRPC기반

#v0.4 #기대해주세요☺

## Service Interface (진행중)



HTTP/  
Websocket



Admin



Developer



Other system/service

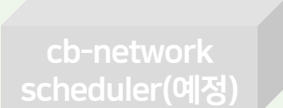
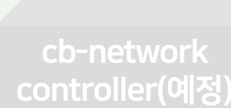
- ✓ APIs 제공을 위한 gRPC 뼈대 마련
- ✓ RESTful APIs 지원을 위한 gRPC Gateway 적용

<https://github.com/cloud-barista/cb-larva/discussions/141>

Click to see details ☺

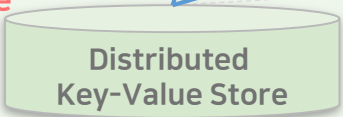
Watch

Protocol Buffer



## cb-network control plane (이벤트 중심 아키텍처)

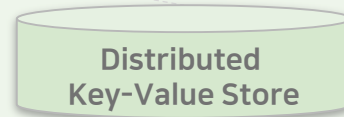
Singapore



Seoul



Seoul



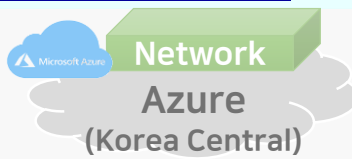
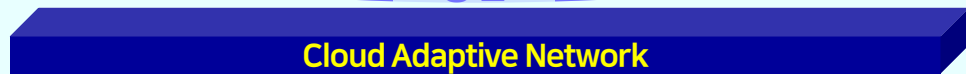
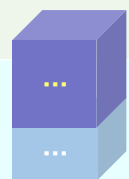
Watch

Watch

Watch

Watch

Watch



## cb-network data plane (메시 토폴로지)

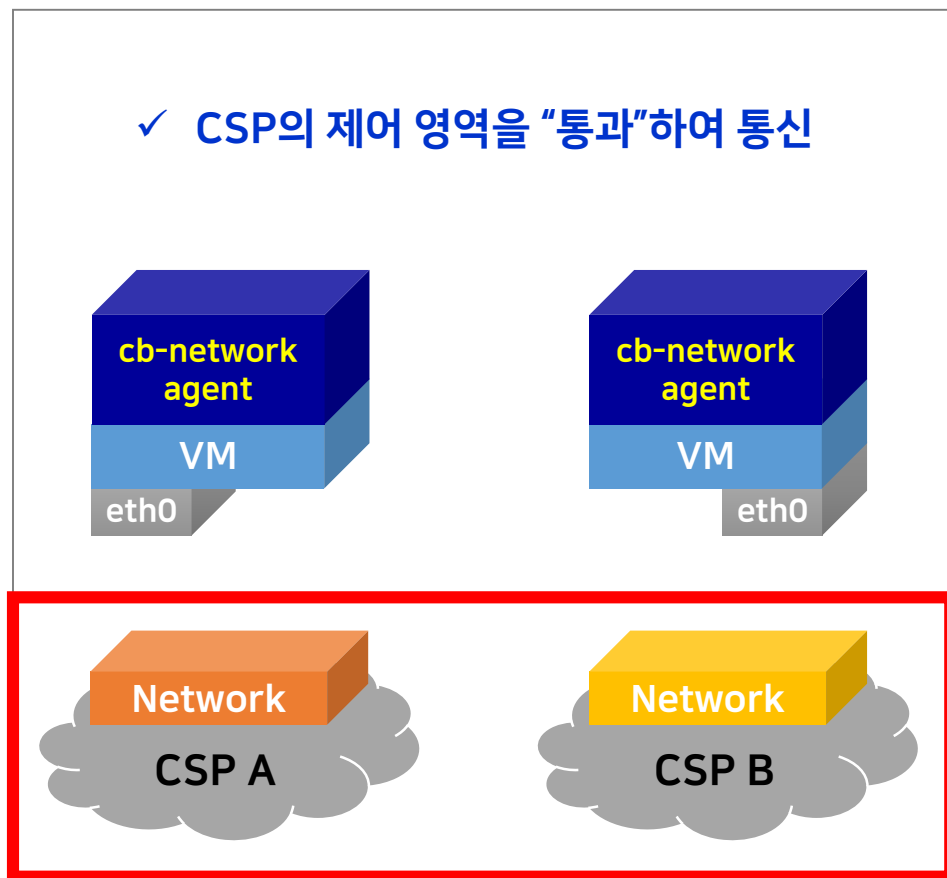
## CSPs plane



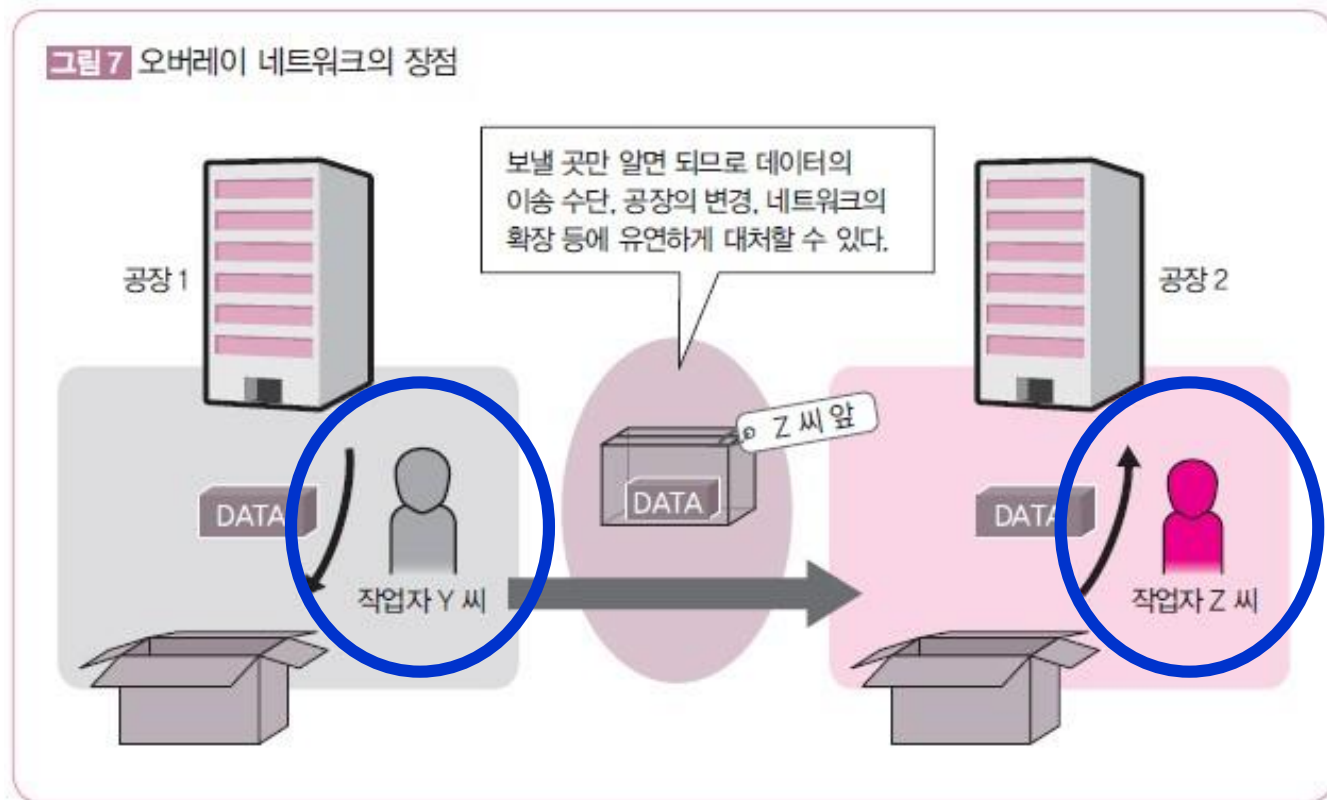
# (주요기술) cb-network agent

(작업자 Y 씨, Z 씨 역할) **Tunneling**을 수행하기 위한 필수 소프트웨어

(Motivated by OpenVPN, Tunneling, DHCP서버, NAT, Routing table)



CSP의 제어 영역 - 서로 다른 Overlay Network가 적용 되어 있음

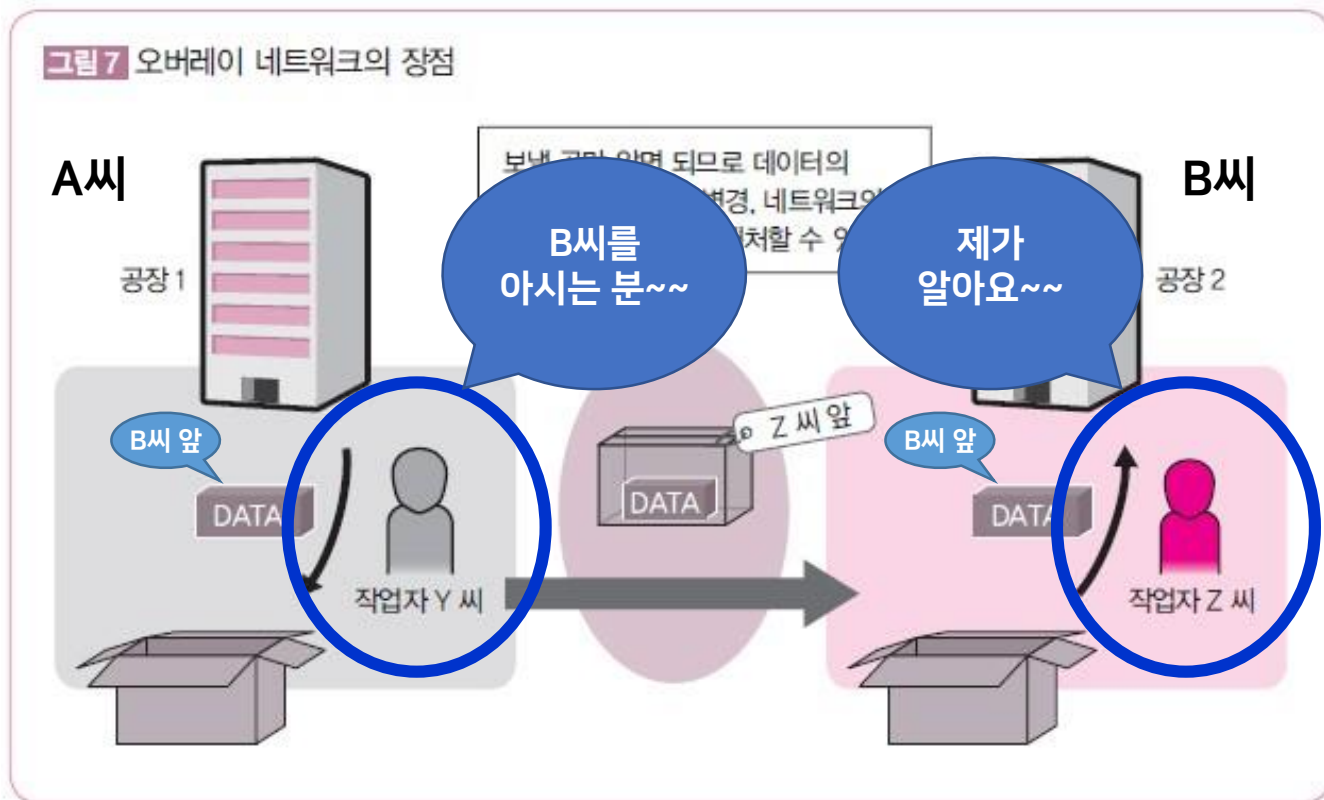
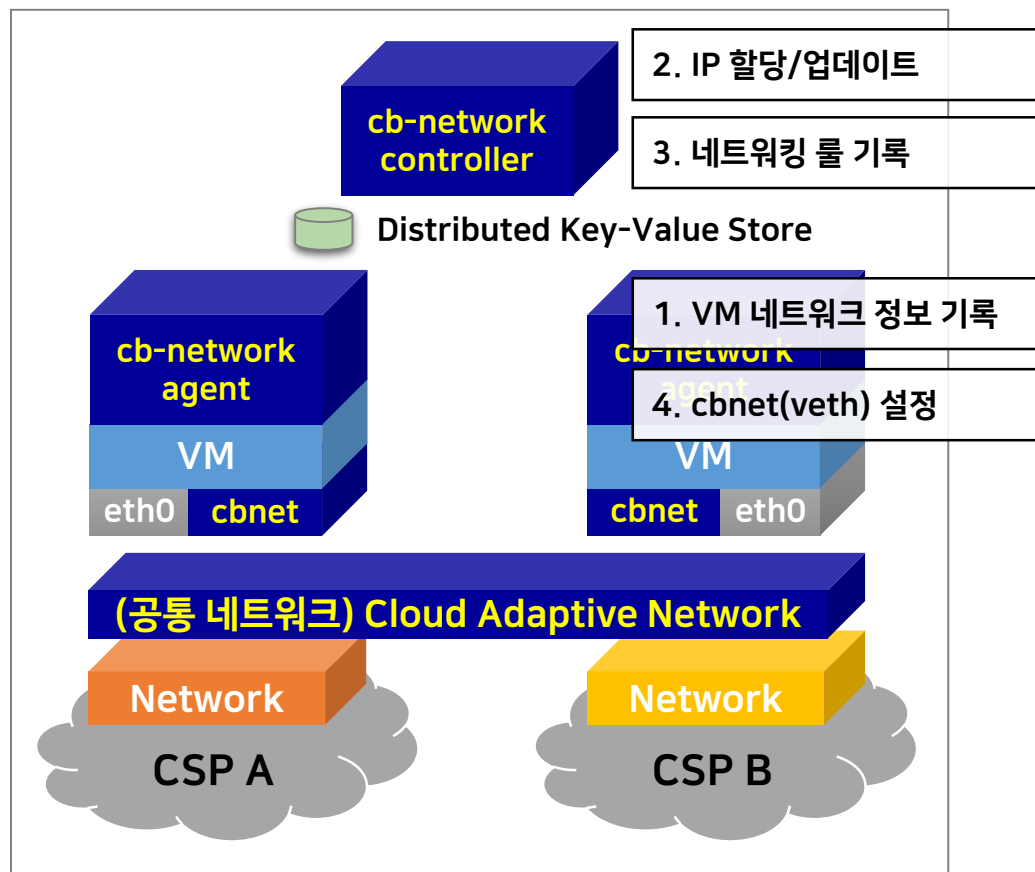


출처: Takaoka Susumu, Takazoe Osamu 공저, 이영란 역 저, 정보문화사, 집에서 배우는 가상화의 기본 개념, 2016

# (주요기술) cb-network controller

(작업자 Y, Z씨를 위한 목적지 정보 관리) **공통 네트워크 구성 및 네트워킹 룰 "사전" 공유/동기화**

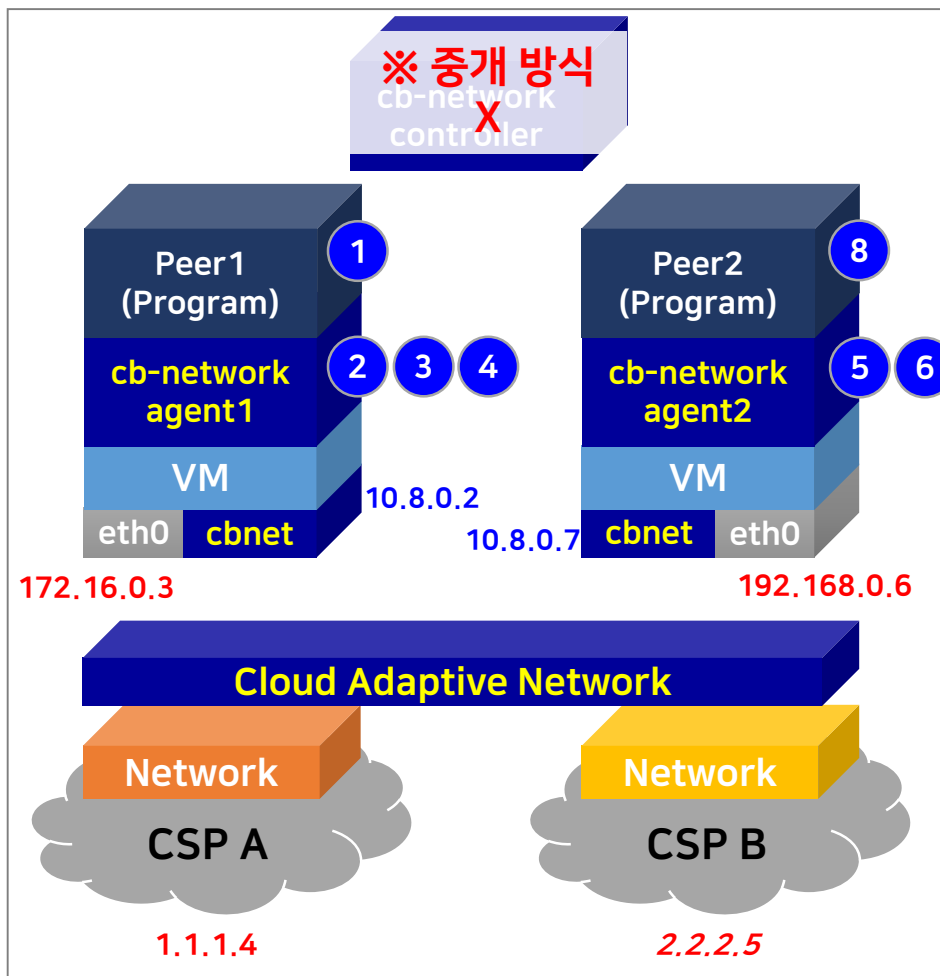
(Motivated by OpenVPN, Tunneling, DHCP서버, NAT, Routing table)



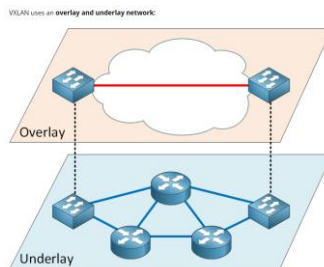
출처: Takaoka Susumu, Takazoe Osamu 공저, 이영란 역 저, 정보문화사, 집에서 배우는 가상화의 기본 개념, 2016

# (예시) Cloud Adaptive Network를 통한 Peer간 통신

메쉬(Mesh)형 토폴로지 채택으로 Peer간 통신 시 중개 노드의 위치에 따른 성능 차이를 최소화



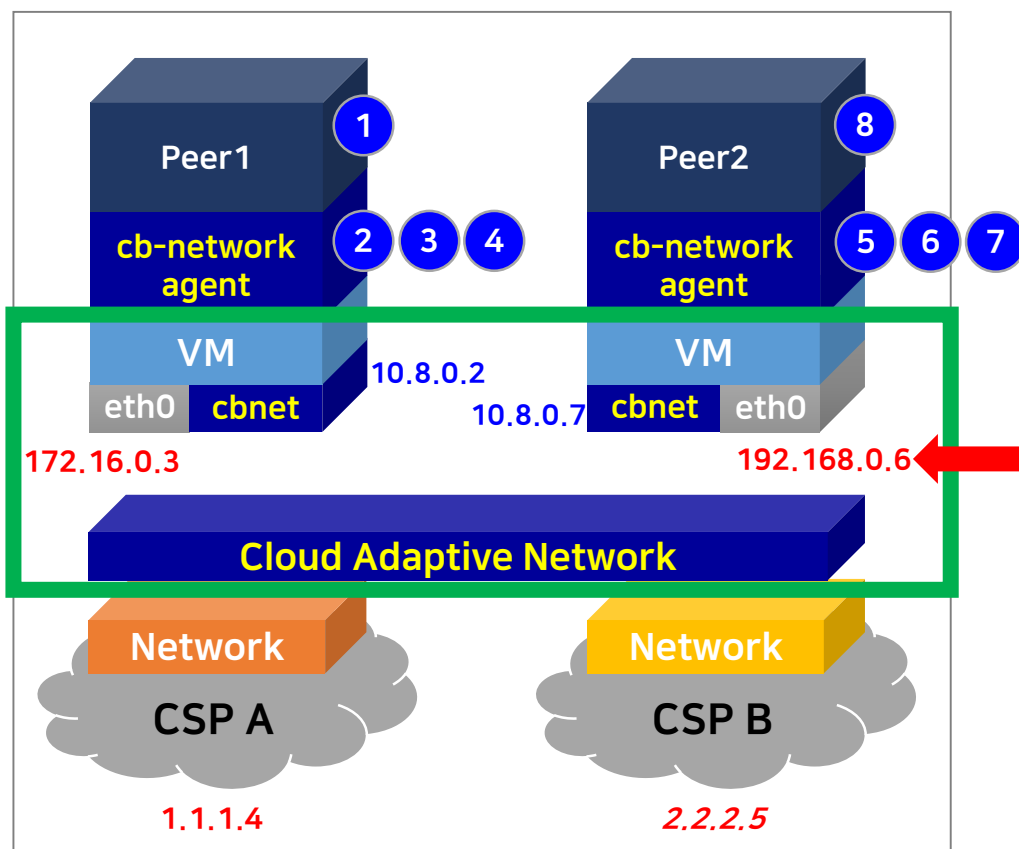
※ 기존 프로그램  
그대로 사용 O



## Peer간 통신 Workflow

- 1 Peer1: Peer2(10.8.0.7)에게 메시지 전송
- 2 Agent1: 메시지 감지(Read)
- 3 Agent1: Networking Rule 조회
- 4 Agent1: 메시지 포장(Encap. to 2.2.2.5)후 전송
- 5 Agent2: 메시지 감지(Read)
- 6 Agent2: 메시지 포장 해제(Decap.)
- 7 Agent2: 메시지를 cbnet으로 전송
- 8 Peer2: 메시지 수신

Q. (공통 네트워크) Cloud Adaptive Network의 주소 공간은 어떻게 할당 하죠 ?



VM이 속한 주소 공간(172.16.0.0, 192.168.0.0)은 다 다른데...

10.8.0.2와 10.8.0.7이 포함된 주소 공간은?

네트워크 관리자가???  
시스템이 알아서???

# Idea에 대해서

글로벌 스케일 네트워킹의 고려사항  
(Introduced at 3차 컨퍼런스)

엉뚱한 Idea?!



- ✓ 주소 공간
- ✓ Address space
- ✓ Address range
- ✓ CIDR block
- ✓ CIDR range





# (예시) 가상머신의 호스트 네트워크는 ...

hermitkim1 commented on 16 Sep

What would you like to be added?  
: I like to add a routing rule for each VM.

Why is this needed?  
: It is needed to connect each VM to the host network, which is various.

[Cases in each VM]

AWS VM

```

ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
    inet 192.168.2.81 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::9d:8fff:fe5e:d292 prefixlen 64 scopeid 0x20<link>
    eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    RX packets 74 bytes 16794 (16.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 825 bytes 91320 (91.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
  
```

192.168.2.81/24

MS Azure VM

```

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.3.4 netmask 255.255.255.0 broadcast 192.168.3.255
    inet6 fe80::20d:3aff:fe32:9711 prefixlen 64 scopeid 0x20<link>
    eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    RX packets 1084 bytes 1120153 (1.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4212 bytes 1120153 (1.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
  
```

192.168.3.4/24

GCP VM

```

ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1460
    inet 192.168.4.2 netmask 255.255.255.255 broadcast 0.0.0.0
    inet6 fe80::4001:c0ff:fea8:402 prefixlen 64 scopeid 0x20<link>
    eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1460
    RX packets 1021 bytes 131501 (131.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1021 bytes 131501 (131.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
  
```

192.168.4.2/32

✓ CSP의 Routing rule을 따르면 X

## (예시) Routing Table

route

\$ sudo route -n

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.0.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
0.0.0.0	192.168.0.1	0.0.0.0	UG	0	0	0	eth0

netstat

\$ netstat -rn

Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
192.168.0.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
0.0.0.0	192.168.0.1	0.0.0.0	UG	0	0	0	eth0

ip

\$ ip route list

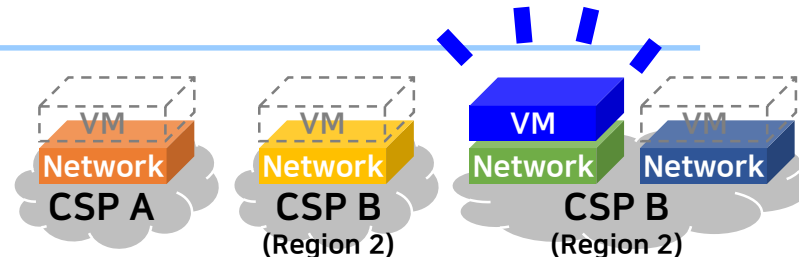
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.103  
default via 192.168.0.1 dev eth0

# (이슈) Cloud Adaptive Network의 주소 공간 할당 전에 ...

## 유동

클라우드 자원은 유동적이다?!

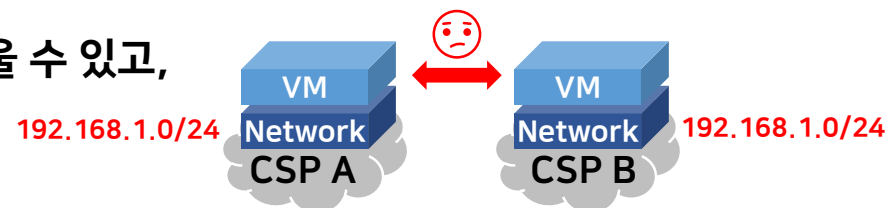
→ 인프라(MCIS) 생성 전에는  
정확한 네트워크 정보를 얻기 어려워요.



## 중복

클라우드간에 중복된 Subnet이 생성 될 수 있다?!

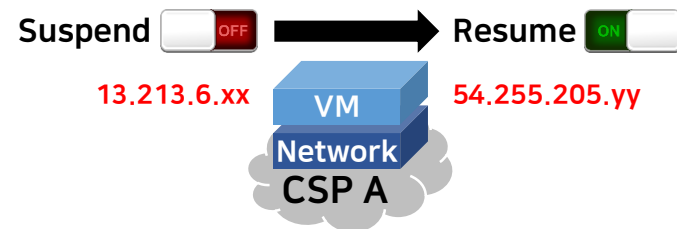
→ VPN Gateway를 활용해도 통신이 어려울 수 있고,  
Supernetting 또한 어려워요.



## 변동

VM관련 네트워크는 변경될 수 있다?!

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



## 규모

글로벌 스케일 인프라?!

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



# 기본으로 돌아가서...



192.168.1 .1 / 24



Network

Host



Prefix

(Subnet Mask)

일하다보니 전공책을  
다시 보게 되더라...



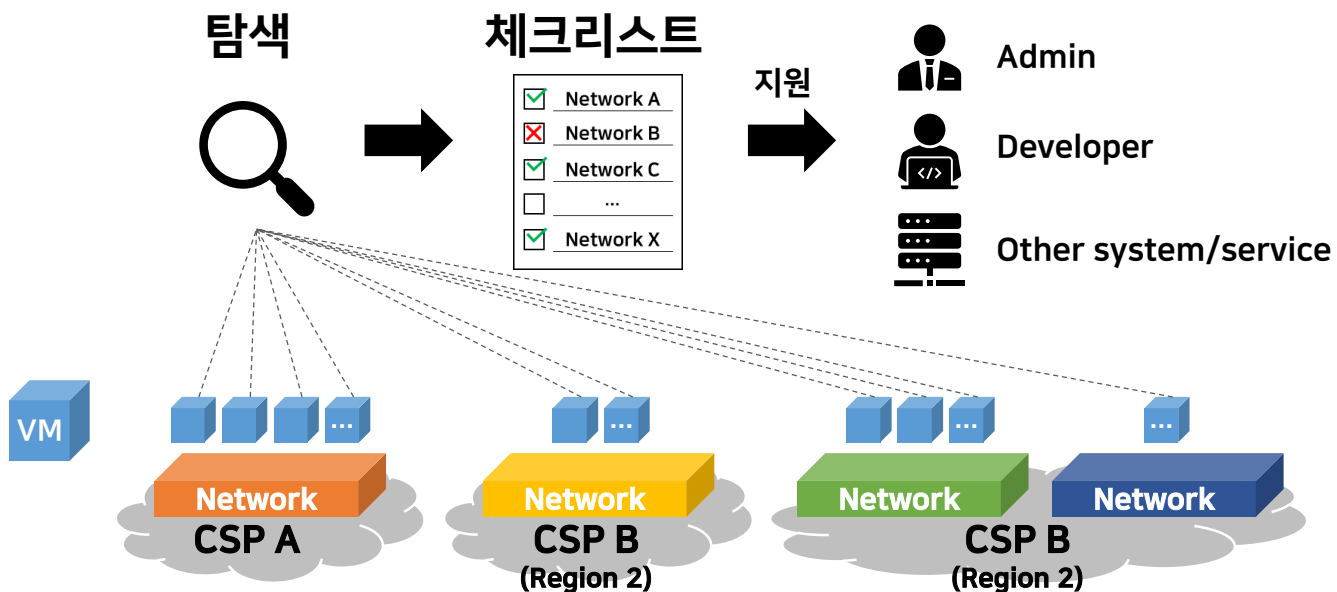
선배님들

- ✓ 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

# (주요기술) 가용 IPv4 private address spaces 지원

(also known as CIDR block, CIDR range, IP address range)

(Idea) 모든 호스트의 CIDR 접두사를 탐색하여 가용한 IPv4 private address spaces 지원



## 구조체

```
type AvailableIPv4PrivateAddressSpaces struct {
    RecommendedIPv4PrivateAddressSpace string
    AddressSpace10s                     []string
    AddressSpace172s                    []string
    AddressSpace192s                    []string
}
```

하나의 Prefix에 여러 네트워크(netid)가 포함됨 (향후 개발 사항)

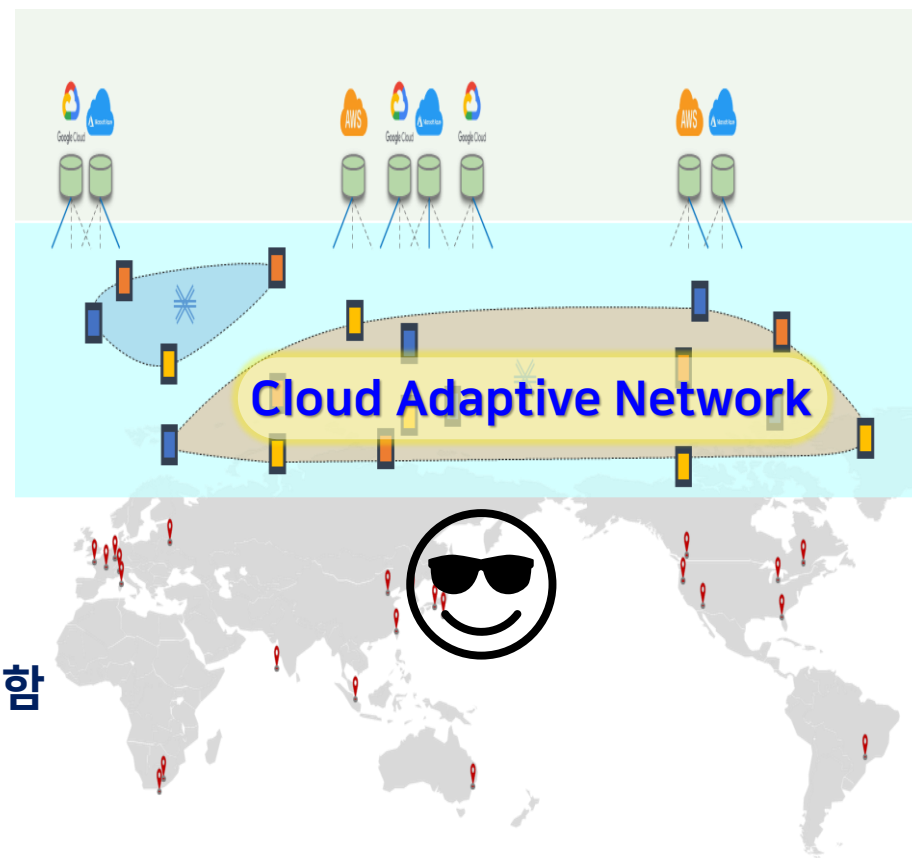
(예시) [All 128 of the Possible /23 Networks for 192.168.\*.\*]

Network Address	Usable Host Range	Broadcast Address
192.168.0.0	192.168.0.1 – 192.168.1.254	192.168.1.255
192.168.2.0	192.168.2.1 – 192.168.3.254	192.168.3.255
192.168.4.0	192.168.4.1 – 192.168.5.254	192.168.5.255
192.168.6.0	192.168.6.1 – 192.168.7.254	192.168.7.255
192.168.8.0	192.168.8.1 – 192.168.9.254	192.168.9.255
...	...	...
192.168.252.0	192.168.252.1 – 192.168.253.254	192.168.253.255
192.168.254.0	192.168.254.1 – 192.168.255.254	192.168.255.255

# (이슈 해결) Cloud Adaptive Network 할당 가능!

지원 / 추천된 주소 공간을 활용하여, **Cloud Adaptive Network 할당**

- 유동** Resolve → 인프라의 네트워크 정보를 바탕으로
- 중복** Resolve → 중복 될 수 있는 서로 다른 서브넷 상에서
- 변동** Resolve → 변화에 동적으로 적응하고
- 규모** Resolve → 대규모 인프라/응용들이 공통 네트워크에 존재하는 것으로 인식함



# 공통 네트워크 기반을 마련하면?!

우리의 서비스

#글로벌

#런칭 #테스트

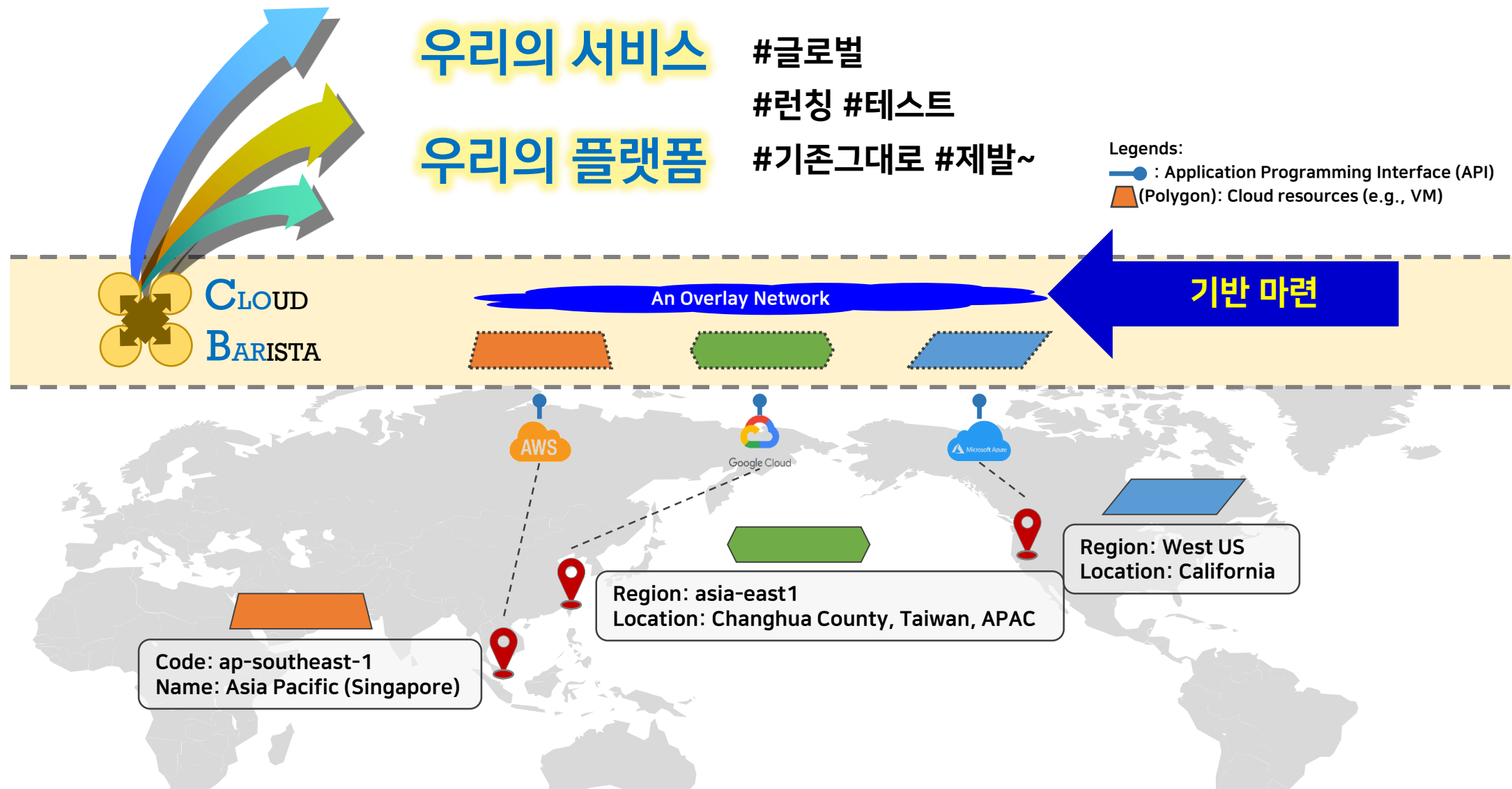
우리의 플랫폼

#기존그대로 #제발~

Legends:

—●— : Application Programming Interface (API)

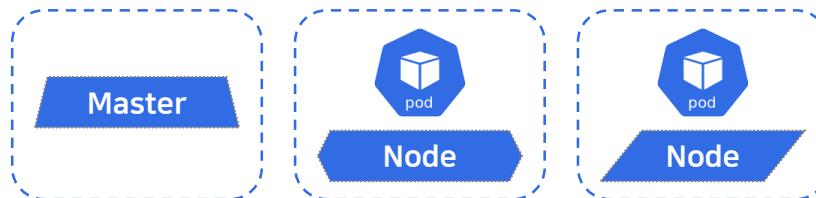
▭ (Polygon): Cloud resources (e.g., VM)





# (유스케이스) 멀티클라우드에 단일 Kubernetes (K8s) 클러스터 구축

조금은 다른 관점이지만..., 네트워크 정상 동작 확인용 ^^;;



Legends:

—●— : Application Programming Interface (API)

▭ (Polygon): Cloud resources (e.g., VM)

M: Master N: Node

## 클라우드바리스타 컴포넌트간 연관성



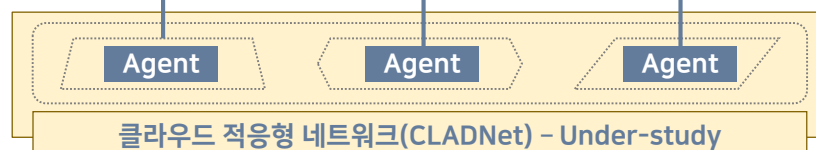
### CB-Tumblebug

멀티클라우드 인프라 서비스  
통합 운용/관리

### CB-Spider

멀티클라우드 인프라 연동

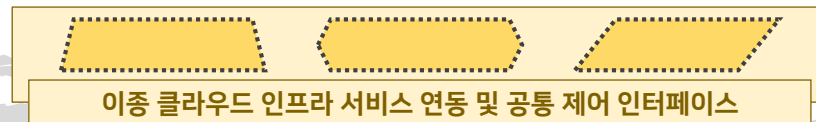
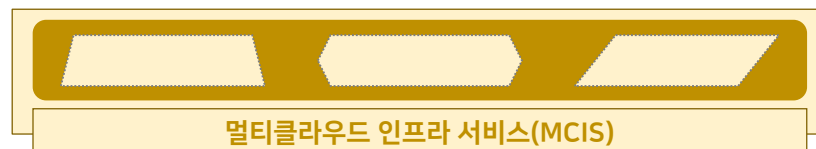
An Overlay Network



### CB-Larva

클라우드바리스타 실험실

Cloud-Barista에  
새로운 니즈 수용하기



Code: ap-southeast-1  
Name: Asia Pacific (Singapore)

Region: asia-east1  
Location: Changhua County, Taiwan, APAC

Region: West US  
Location: California

Click to see details ☺

# 시연을 위한 클라우드바리스타 컴포넌트 배치 및 활용



or

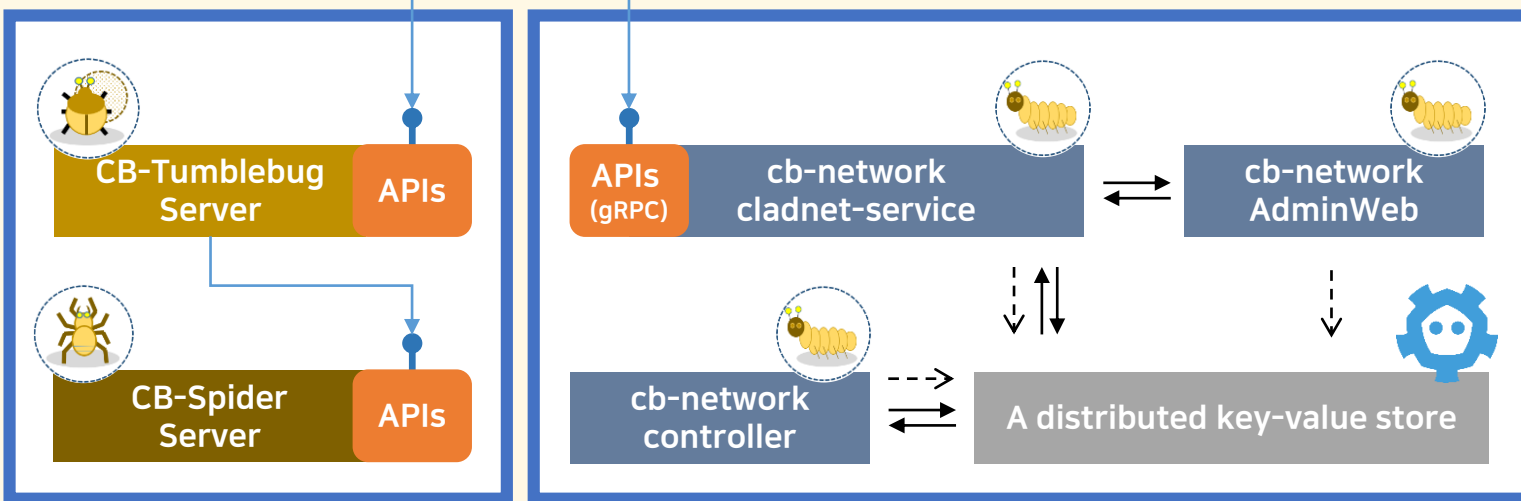


Client program  
(as a user)

1. Create an MCIS

3. Configure a CLADNet

2. Create a CLADNet address space



## 클라우드바리스타 컴포넌트

Legends:

MCIS: Multi-Cloud Infra Service

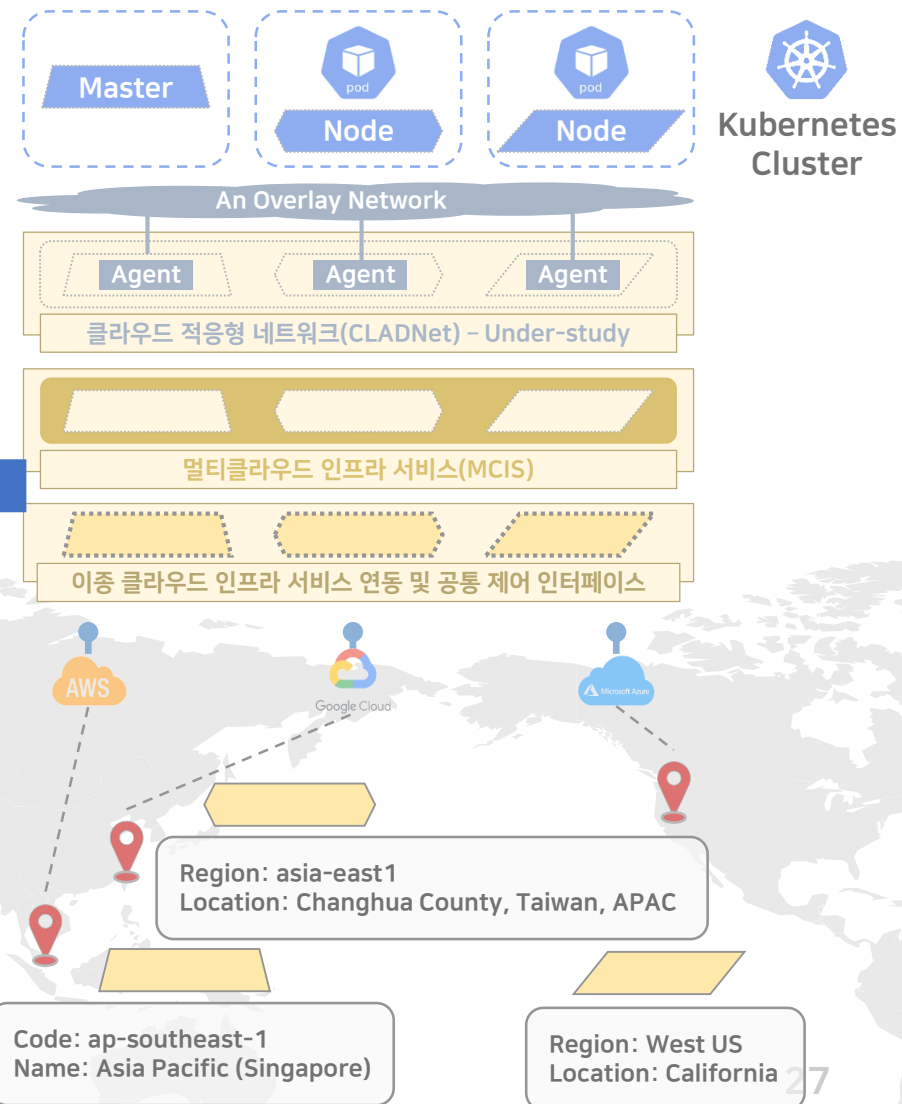
CLADNet: Cloud Adaptive Network

—●— : Application Programming Interface (API)

▭ (Polygon): Cloud resources (e.g., VM)

→ : Request/response

- -> : watch



## 멀티클라우드에 기존 서비스를 올리기 위한 첫걸음

Click to see A-Z video ☺

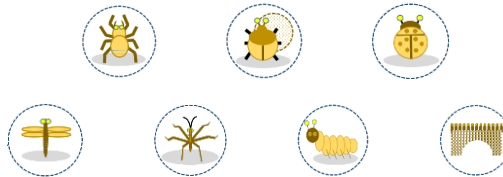
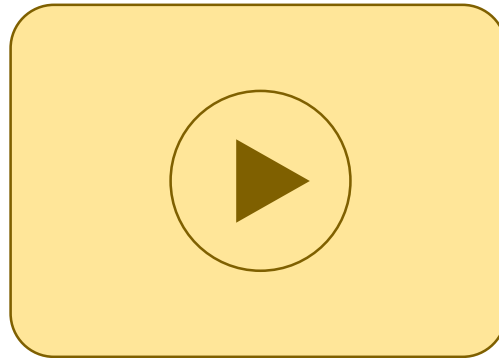
<https://drive.google.com/file/d/1GFuPe-s7IUCblfLAv-Jkd8JaiQci66nR/view?usp=sharing>

관련 영상은 Cloud-Barista 커뮤니티 YouTube 채널에 업로드할 예정 입니다 ^^

 YouTube : Cloud-Barista 커뮤니티(<https://cloud-barista.github.io/youtube>)

# 멀티클라우드에 기존 서비스를 올리기 위한 첫걸음

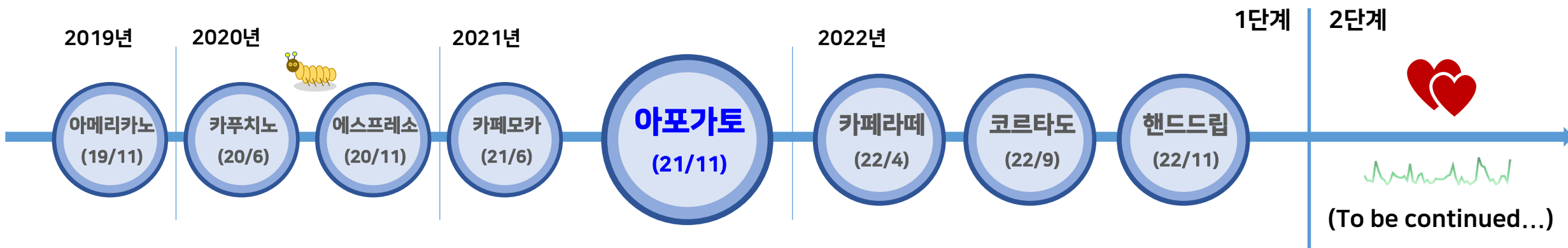
동영상 시연이 준비되어 있습니다.



세션을 놓치셨다면? 걱정하지 마세요! Cloud-Barista 커뮤니티 YouTube 채널에서 확인하실 수 있습니다 ^^

 YouTube : Cloud-Barista 커뮤니티(<https://cloud-barista.github.io/youtube>)

# 연구 개발 현황 및 향후 계획(비전)



완료    진행    예정

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

Cloud Adaptive Network 프로토타입 개발

(주요 포인트: 성능 개선) Cloud Adaptive Network 주요 기능 개발, 테스트 및 고도화

(외부 연계) Cloud Adaptive Network 서비스 제공 및 프레임워크간 연계

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)



# (공개SW) 여러분의 관심을 먹고 살아요 ^^

정보, 문서, 코드, 리뷰, 의견 하나 하나가 기여이고 이를 통해 발전합니다

cloud-barista / cb-larva Public

Code

Issues

Pull requests

Discussions

Actions

Projects

Wiki

Security

Insights

main

2 branches

5 tags

Go to file

Add file

Code

hermitkim Merge pull request #146 from hermitkim/hermitkim1

204 commits

github

poc-cb-net

scripts/Kubernetes

gitignore

golangci.yml

BRANCHING-AND-RELEASING...

CODE\_OF\_CONDUCT.md

Dockerfile

LICENSE

README.md

go.mod

go.sum

Install golangci-lint by a...

Update template and chang...

Set network env variable fr...

Update template and gitignore

Refactor the source code of...

Prepare for Cafe Mocha release

Create CODE\_OF\_CONDUCT...

Rename the cb-network server to controller

Create LICENSE

Prepare for Cafe Mocha release

Update README

Lay the foundations for APIs

Lay the foundations for APIs

About

CB-Larva is Cloud-Barista Incubator. The incubator researches and develops essential technologies for multi-cloud.

insulator multi-cloud cb-larva cb-network cb-subnet

Readme

Apache-2.0 License

Releases

A release of MQTT-base...

Packages

Contributors

hermitkim Yunkon (Alvin) Kim

seokho-son Seokho Son

Languages

Go 35.9%

Shell 11.0%

JavaScript 6.5%

Dockerfile 0.6%

CSS 53.8%

HTML 11.3%

Makefile 0.1%

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.

cloud-barista / cb-coffeehouse Public

Code

Issues

Pull requests

Discussions

Actions

Projects

Wiki

Security

Insights

main

2 branches

0 tags

Go to file

Add file

Code

hermitkim Apply the pre-release of CB-Tumblebug

135 commits

github/workflows

docs

scripts

gitignore

README.md

README.md

Run a workflow

Apply the pre-release

Update golang in

Update golang in

Update README

contrib-readme

About

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

Readme

Contributors

Languages

Shell 100.0%

다른 언어로 읽기: English or 한국어.

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 Penny가량의 커피 한잔을 마시며 지식인들이 다양한 의견을 공유하는 사교 클럽이었습니다.

이처럼 Cloud-Barista's Coffeehouse에서 다양한 정보를 "편하게" 공유하셨으면 좋겠습니다.

자유롭게 정보, 설명, 의견을 공유하시면서 오픈소스 프로젝트에 참여하시고, 자연스럽게 Cloud-Barista의 여러 Repository에서 기여 포인트를 찾으시면, Contributor, Reviewer, Maintainer로 거듭나실것입니다!

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 (GHCRI)

GitHub Actions

Golang CI/CD (Continuous Integration/Continuous Delivery)

Cloud-Barista 이슈 및 해결 방안

설치/배포 자동화 Script

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

미래 컨트리뷰터를 위한 메시지

설명/정보 공유 가이드

이것은 불필요한 중복 기여를 완화하기 위한 대략적인 가이드입니다.

소스코드/스크립트 공유

Issues 탭에서 Issue를 생성 (제목을 명확하게 적어주세요.)

Branch를 생성

2021 오픈소스 컨트리뷰션 아카데미

Contribution Academy

미래 컨트리뷰터를 위한 각 클라우드별 VM 생성 가이드

고현경

남기백

박범수

박소연

박인호

손현준

안태건

유재혁

이도훈

이장훈

임승경

조성빈

최수현

최지현

Learning about the public cloud

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



# Topics

Multi-cloud network

Multi-cloud storage

Event-driven architecture

Micro-service architecture

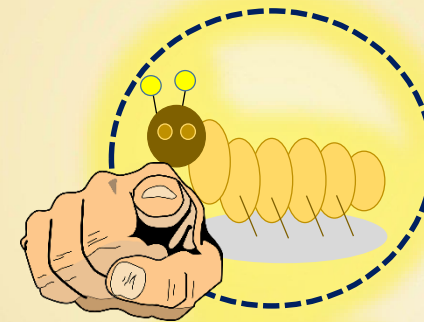
Golang

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/J0wmFLMxc1w>

# (참고) Cloud Adaptive Network와 MCKS CNI의 관계

요약: 유사해 보일 수 있으나, 적용 도메인/레이어가 다르기에 도전과제 또한 상이함

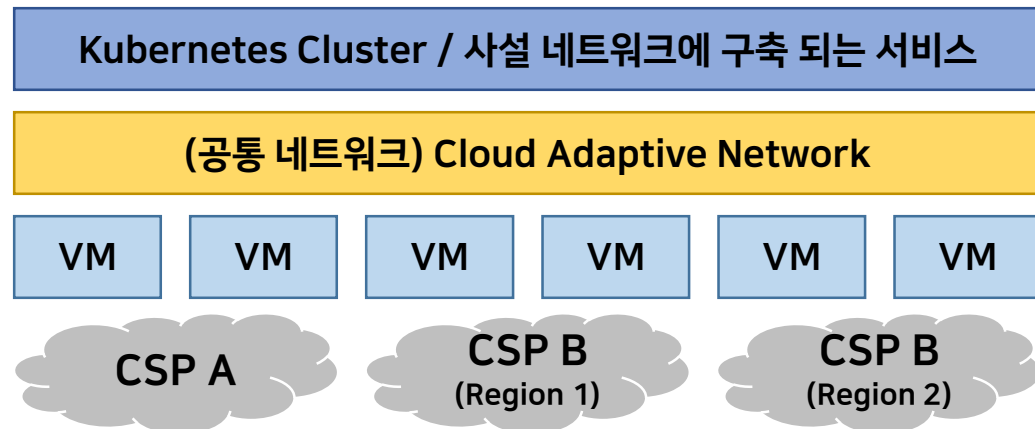
## Cloud Adaptive Network 상의 Kubernetes Cluster

### [목적]

- ✓ 멀티클라우드 CSP 네트워크의 상이함과 가변성 이슈 해결을 위한 공통 네트워크 서비스 제공

### [활용]

- ✓ 사설 네트워크 내에서 설치 및 운용되던 기존 서비스 배포
- ✓ Kubernetes 뿐만 아니라 다른 서비스 제공 가능 예상



## 멀티클라우드 상의 Multi-Cloud Kubernetes Service (MCKS)

### [목적]

※ 전제조건: 공인IP 운용을 고려하는 CNI 활용

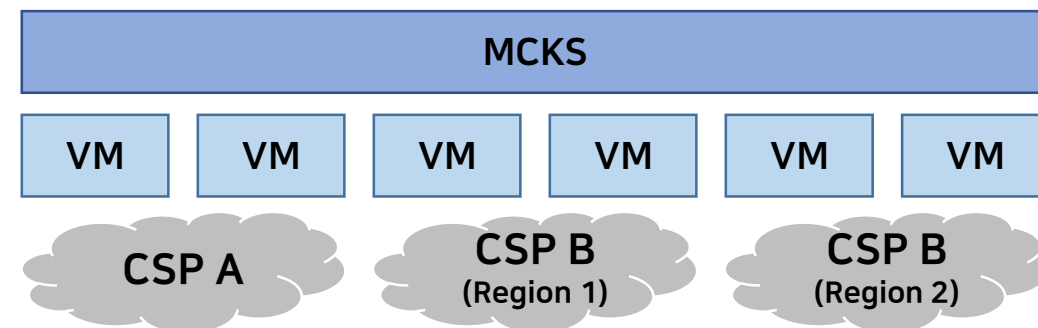
- ✓ 멀티클라우드 상에 Kubernetes 설치 및 운용 이슈 해결을 통한 멀티클라우드 쿠버네티스 서비스 제공

### [활용]

- ✓ 멀티클라우드 워크로드 처리가 가능한 Kubernetes 서비스 제공

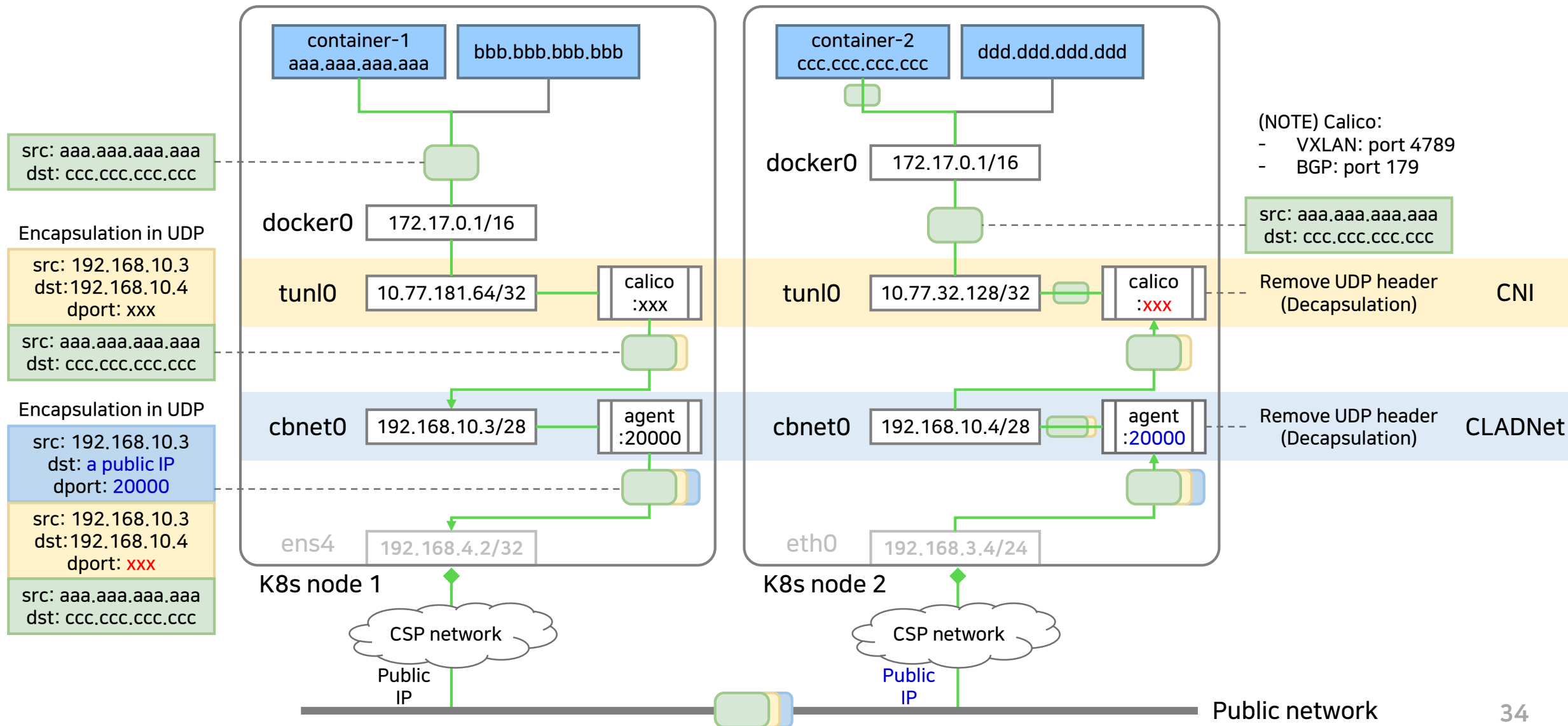
### [이슈]

- ✓ 사설IP 운용만을 고려하는 CNI 활용 어려움



# (참고) Cloud Adaptive Network와 Kubernetes CNI 관계

CLADNet: Cloud Adaptive Network





# 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))

## 멀티클라우드, “새로운 생태계를 향한 클라우드 비긴어게인”

클라우드 바리스타들의 다섯번째 이야기

Cloud-Barista Community the 5th Conference