멀티클라우드, 컴퓨팅 인프라에 제약없는 서비스 생태계

클라우드바리스타 커뮤니티 제7차 컨퍼런스

[CB-Larva]

멀티클라우드 적응형 네트워크

김 윤 곤

CB-Larva 인큐베이터 리더

코르타도(Cortado) 한잔 어떠세요 ?

이번 세션은?

응용/도메인/기관 특화 SW

CLOUD BARISTA

멀티클라우드 서비스 개방형 인터페이스

멀티클라우드 애플리케이션 실행환경
통합관리 프레임워크

멀티클라우드 인프라 서비스
통합 관리 프레임워크

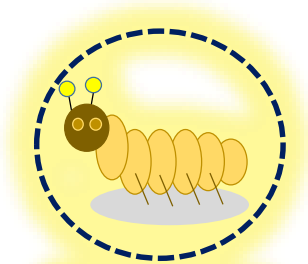
멀티클라우드 인프라 연동
프레임워크

멀티클라우드 인터널 플랫폼

멀티클라우드 서비스 공통 플랫폼

CB-Larva: Cloud-Barista Incubator

멀티클라우드 기술 실험실



CB-Larva

Cloud-Barista의 정체성을 잃지않고,

지속적으로 신규 니즈를 수용하기 위하여

신기술, 부족기술 등의 POC를 수행하며,

이를 Cloud-Barista로 흡수하기 위한 기술 인큐베이터

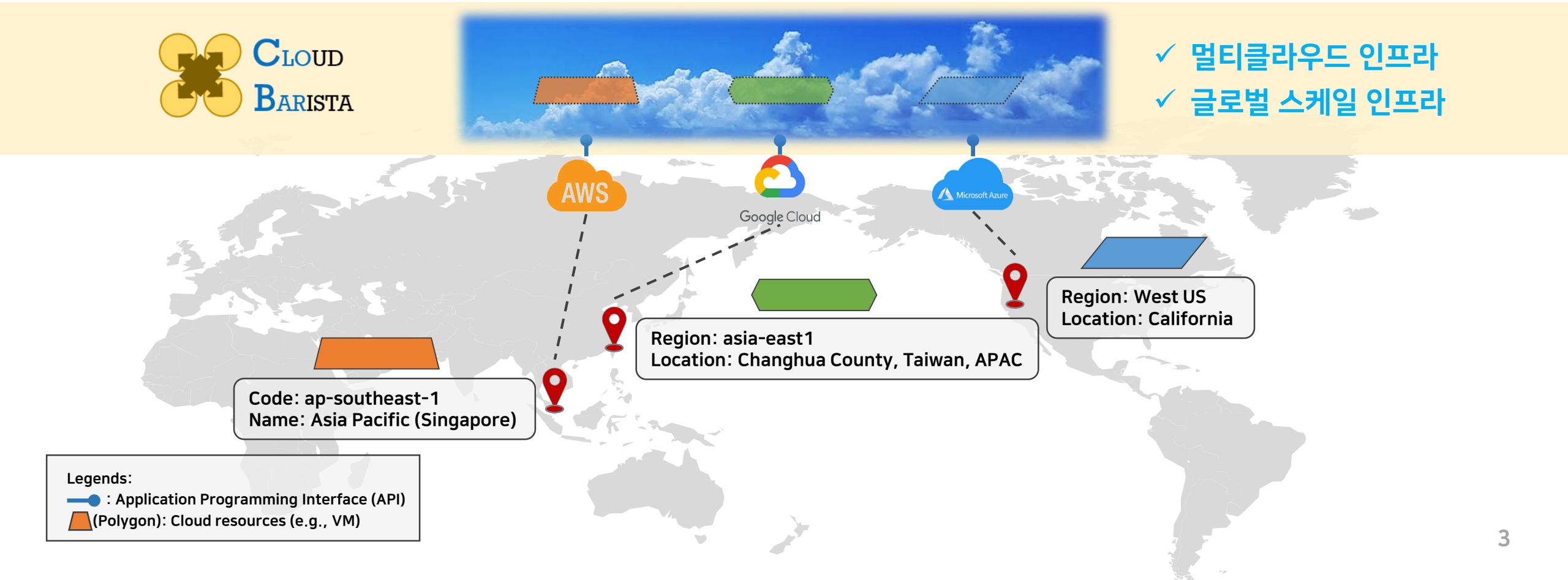
CLOUD BARISTA

목 차

관점을 바꿔서, 하나의 인프라 #글로벌스케일

우리의 서비스 / 우리의 플랫폼

#관점을-바꿔서 #하나의-인프라



- ✓ 멀티클라우드 인프라
- ✓ 글로벌 스케일 인프라

도전과제(Challenge)

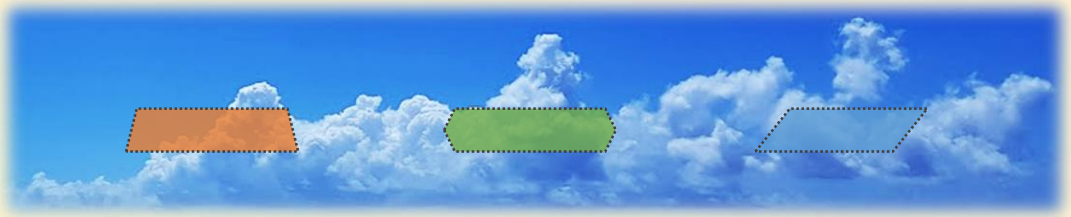
#시작점 #멀티클라우드 #가상 #네트워크

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

openstack. kubernetes ...

튜닝하여 설치
또는
동일 서브넷 구성하여 설치

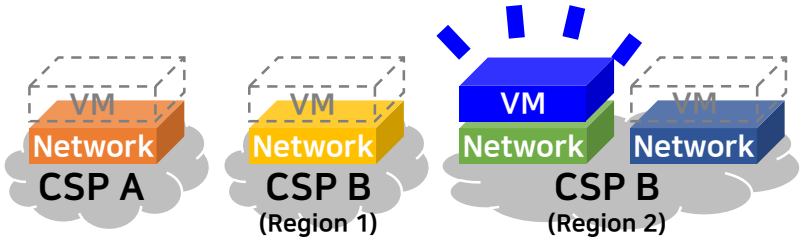


- ✓ 멀티클라우드 인프라
- ✓ 글로벌 스케일 인프라

멀티클라우드의 특성 / 상이함 및 가변성 관련 이슈

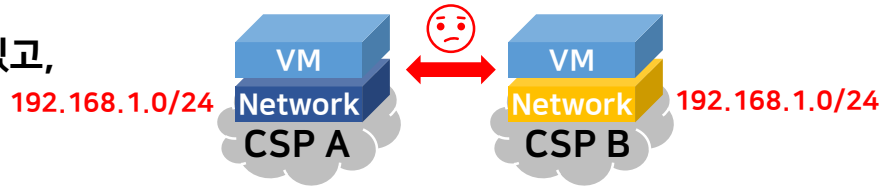
✓ **유동성**
클라우드 자원은 유동적이다?!

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



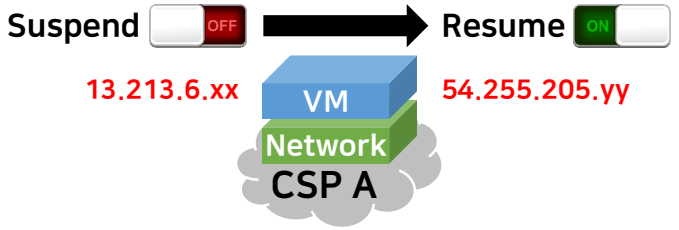
✓ **중복성**
클라우드간에 중복된 Subnet이 생성 될 수 있다?!

➡ 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.

상이함, 가변성에 적응하는 Cloud Adaptive Network

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

clad: 차려입은, 장비한

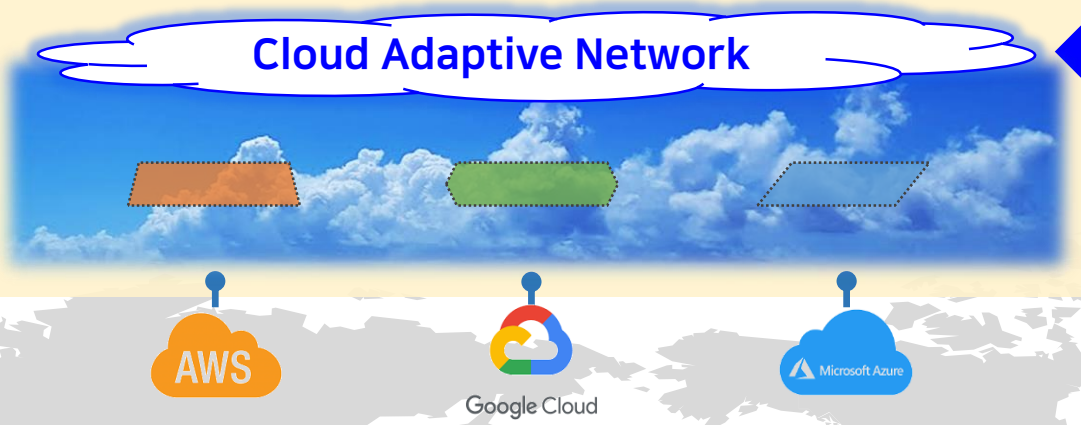
✓ 서로 다른 클라우드의 서로 다른 서브넷 상에서, 인프라 및 응용(VM, Container 등)들이
동일 서브넷에 존재하는 것처럼 사설 IP 기반으로 운용, 관리할 수 있도록 하는 기술

✓ 멀티클라우드의 다양한 네트워크에 적응가능한 오버레이 네트워크로 VM 그룹에 동일 네트워크를 제공함

동일 서브넷 구성하여 설치

openstack. kubernetes

뿐만 아니라,
우리의 응용을 위하여

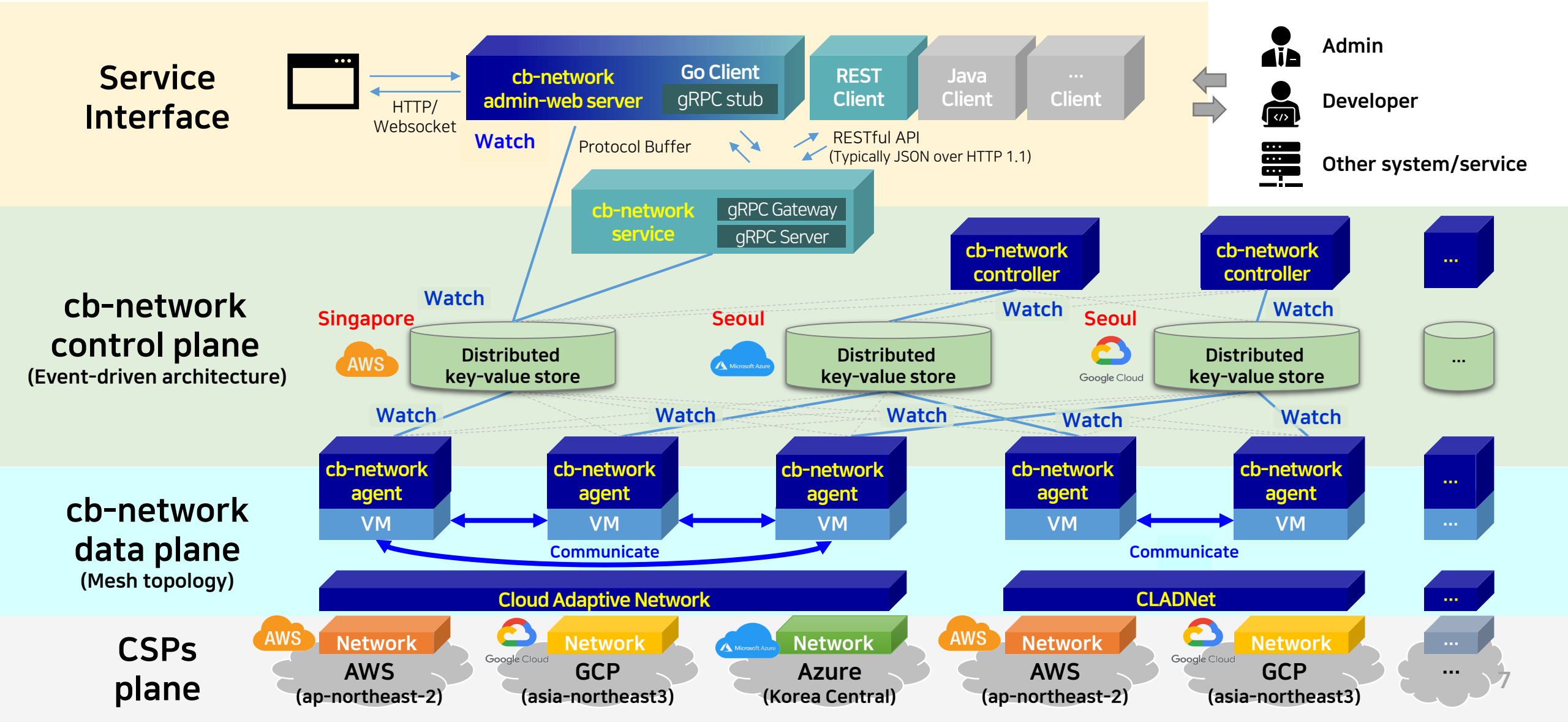


✓ 멀티클라우드 가상/공통 네트워크

✓ 멀티클라우드 인프라
✓ 글로벌 스케일 인프라

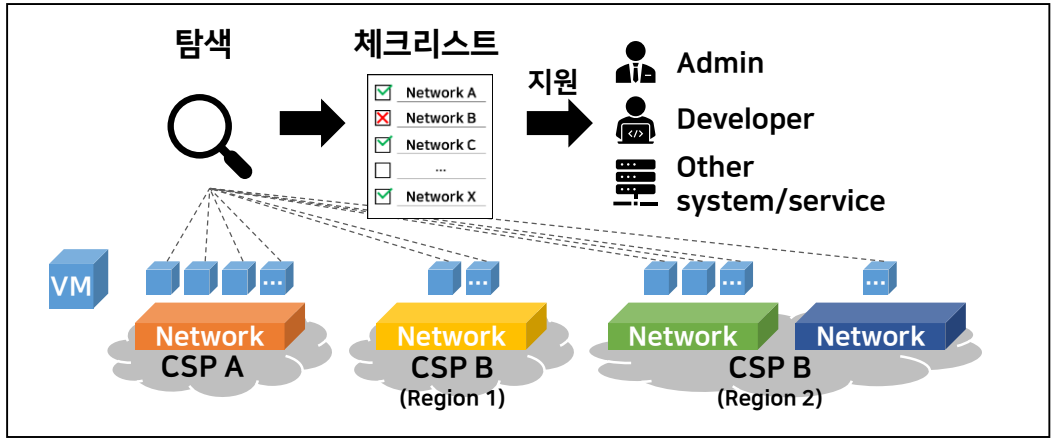
#장애 내성 지향

이벤트 기반 cb-network 시스템의 세부 구조 및 컴포넌트



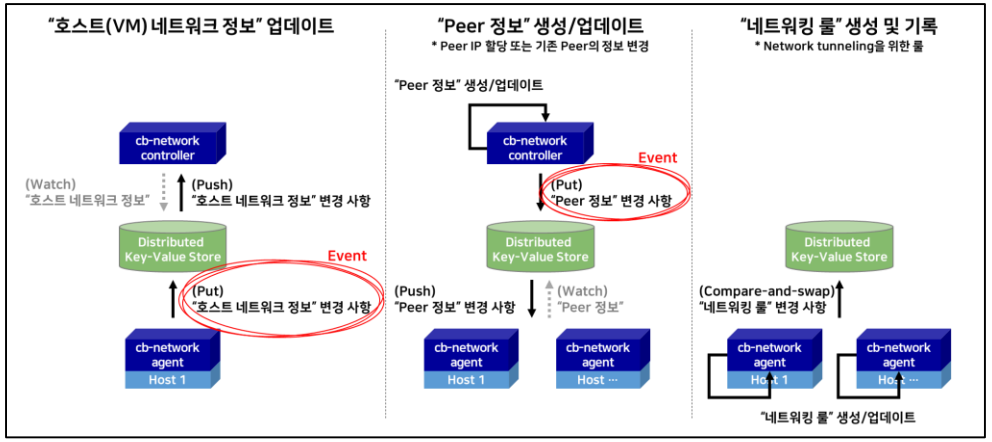
Cloud Adaptive Network의 주요/부가 기능

1. 사설 주소 공간 설정 (사설 주소 공간: private address spaces)



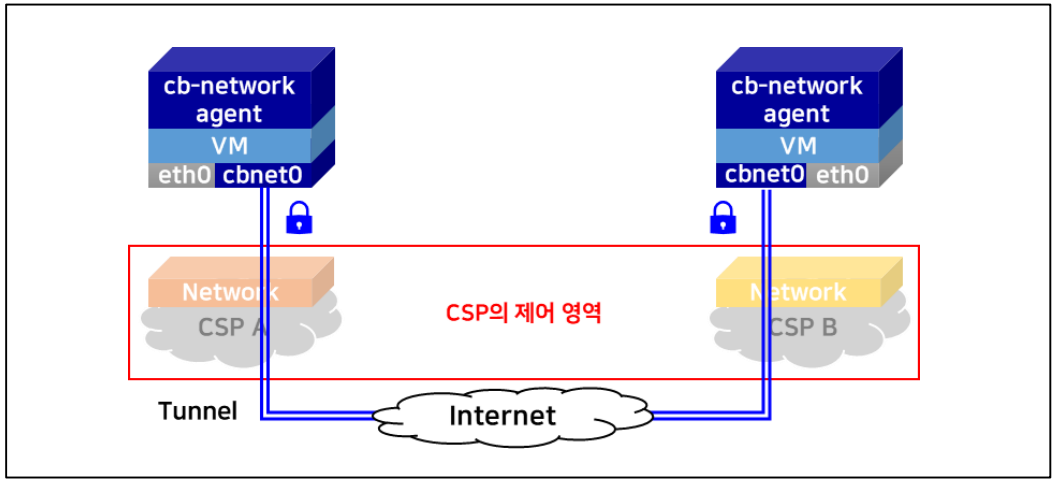
가용한 IPv4 사설 주소 공간 분석 및 추천 서비스 개념도

2. 가상 네트워크 설정(Peer 참여, 네트워킹 룰 사전 할당)



(Event-driven) 가상 네트워크 설정 방법

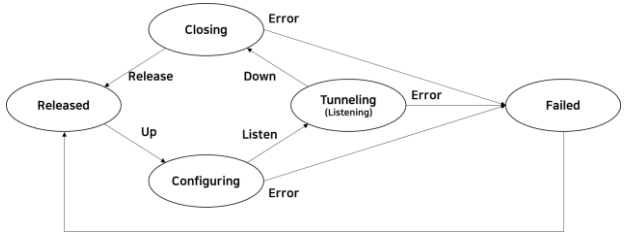
3. 네트워크 터널링을 통한 통신

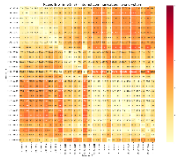


멀티클라우드 가상 네트워크의 터널링 개념도

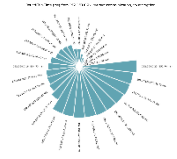
4. 부가 기능

+ Remote-control: up, down, and check-connectivity, etc.
+ State tracking
+ End-to-end encryption: "secret-util"제공(Go package), 기여 가능 RSA Base64
+ Performance evaluation client: 네트워크 변경, Round Trip Time
+ Performance visualizer python





Heatmap



Circular barplot

(API/pkg) 또 하나의 “Adaptive”, cb-network service?!

gRPC + gRPC-Gateway ➡ RESTful API + OpenAPI Specification ➡ Swagger (Supported by SMARTBEAR)

다양한 프로그래밍 언어 및 REST 지원 가능 (1 Port에서 지원중)

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 — Explore

Cloud-Barista Network (cb-network) service 1.0

https://raw.githubusercontent.com/cloud-barista/cb-larva/main/poc-cb-net/docs/cloud_barista_network.swagger.json

Note - `cloud_barista_network.swagger.json` is generated by `openapiv2`

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

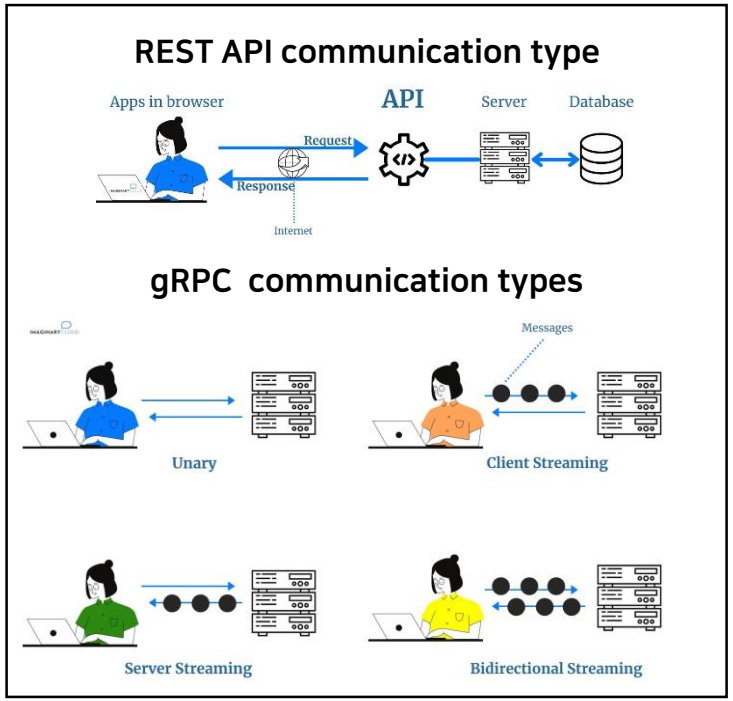
SystemManagementService

- 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

(참고, 장/단점이 존재)

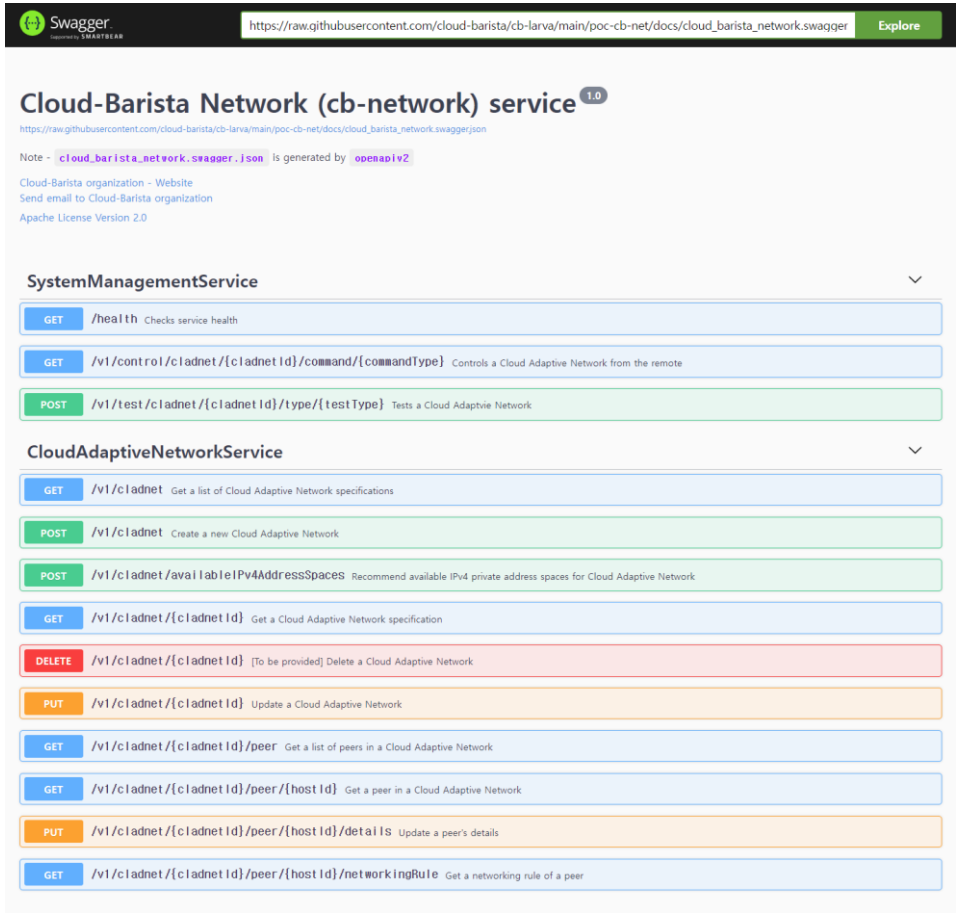


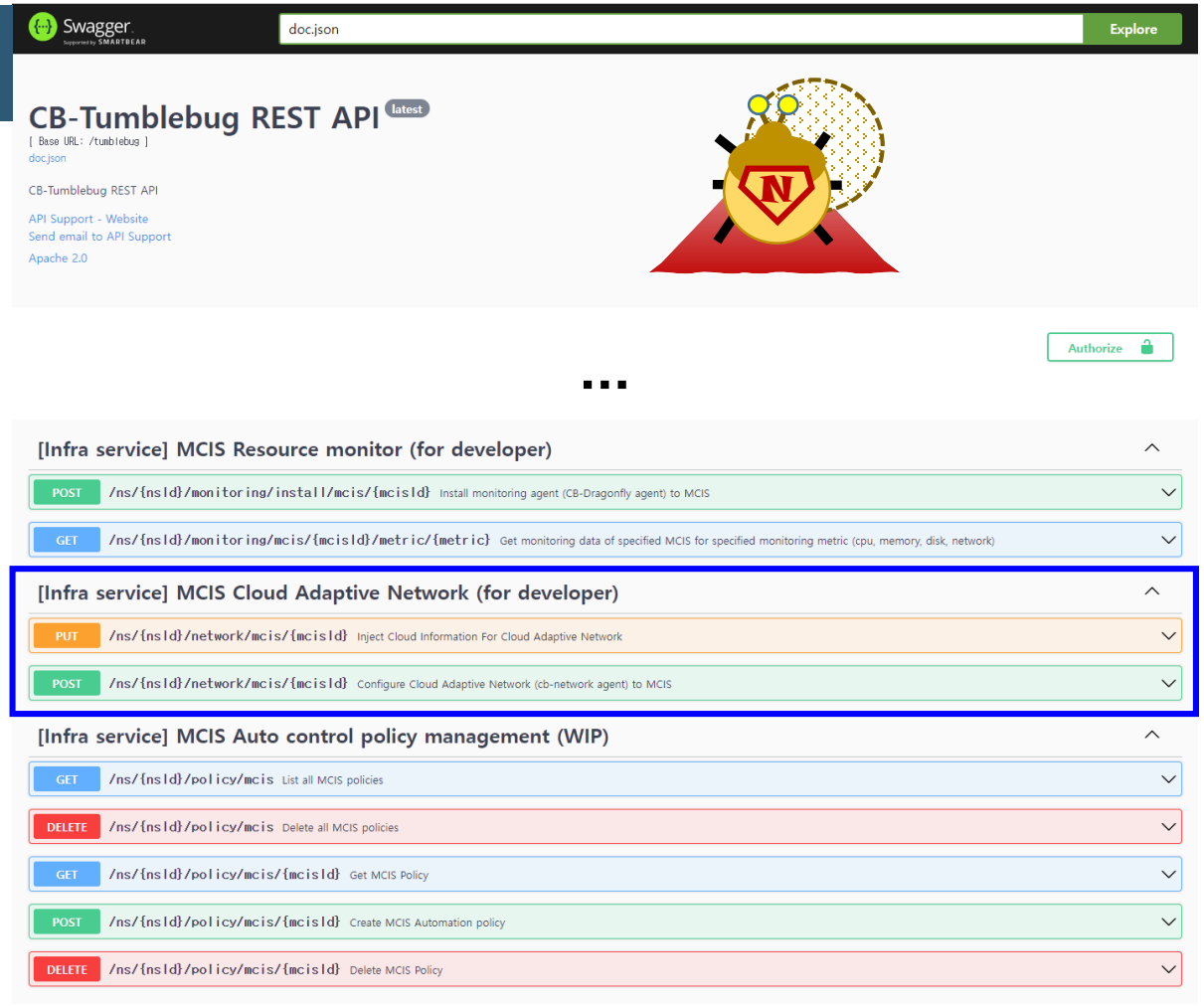
출처: https://www.imaginarycloud.com/blog/grpc-vs-rest/

Cloud Adaptive Network를 입은(Clad) CB-Tumblebug

CB-Tumblebug에 관련 API 추가 (under-development)

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

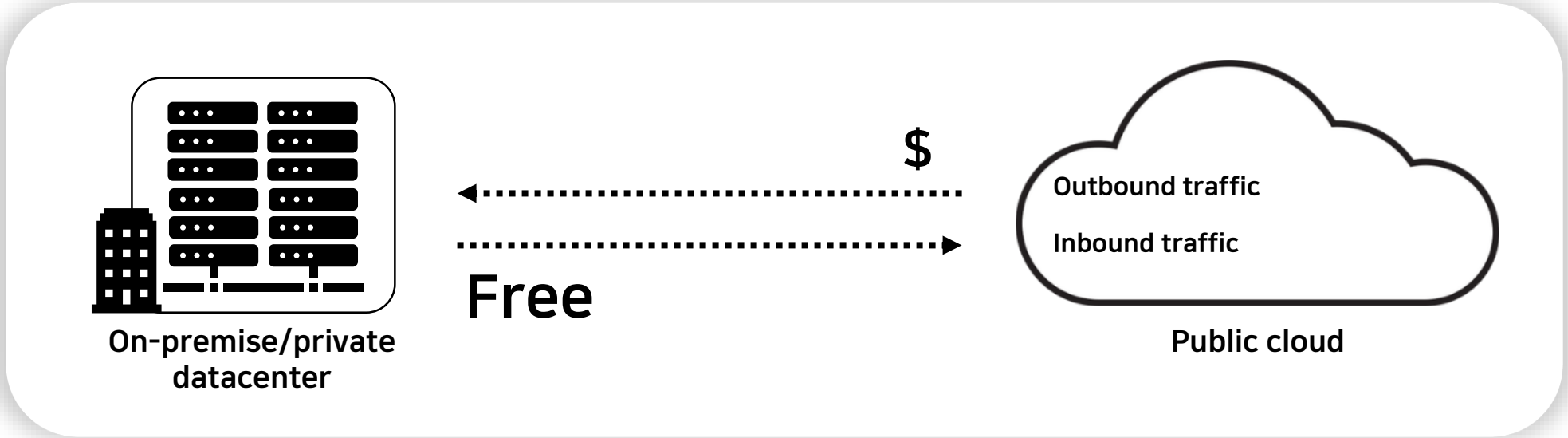




클라우드 인프라의 주요 과금 대상 및 활용 패턴

✓ Compute ✓ **Network** ✓ Storage

(주로 다룰 과금 대상)



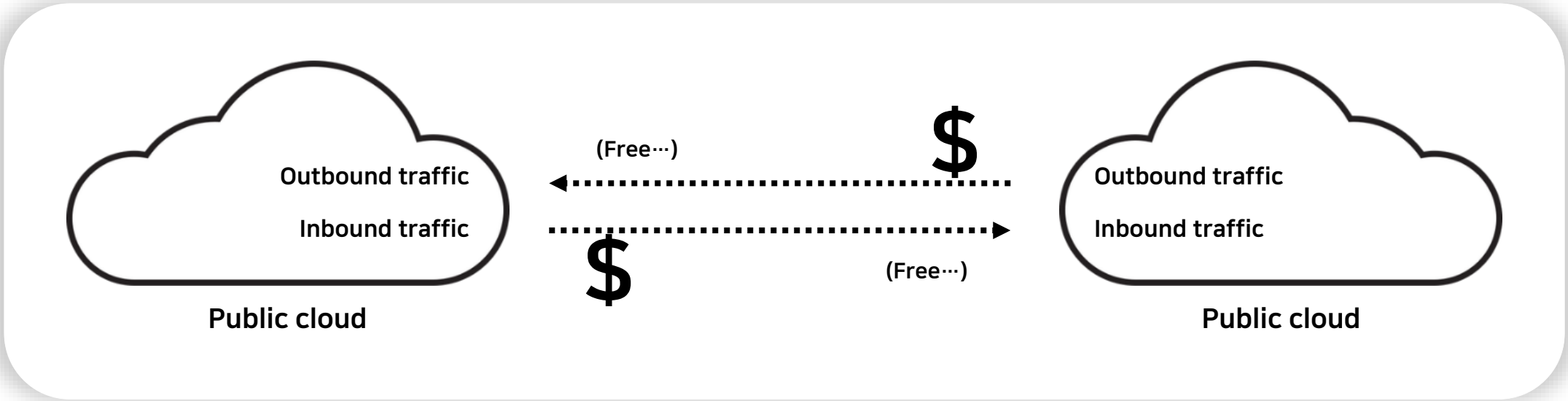
42% of enterprises use the cloud for backup

출처:
Gursimran Singh, “What is Multi-Cloud and Hybrid Cloud | Use-Cases”, 2021 (accessed on 2022-01-27 https://www.xenonstack.com/insights/multi-cloud-vs-hybrid-cloud)

도전과제(Challenge)

#멀티클라우드 #상호작용 #과금

멀티클라우드간에 상호작용 시 과금이 발생함



(사례) Snowflake

멀티클라우드 과금 체계를 잘 파고든 사례 ?

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.

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 @YUNLI626 | 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

Source: CNBC

자세한 내용 - Snowflake documentation 참고: https://docs.snowflake.com/en/index.html

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

동일 AWS Region 상에서 Data Transfer Billing

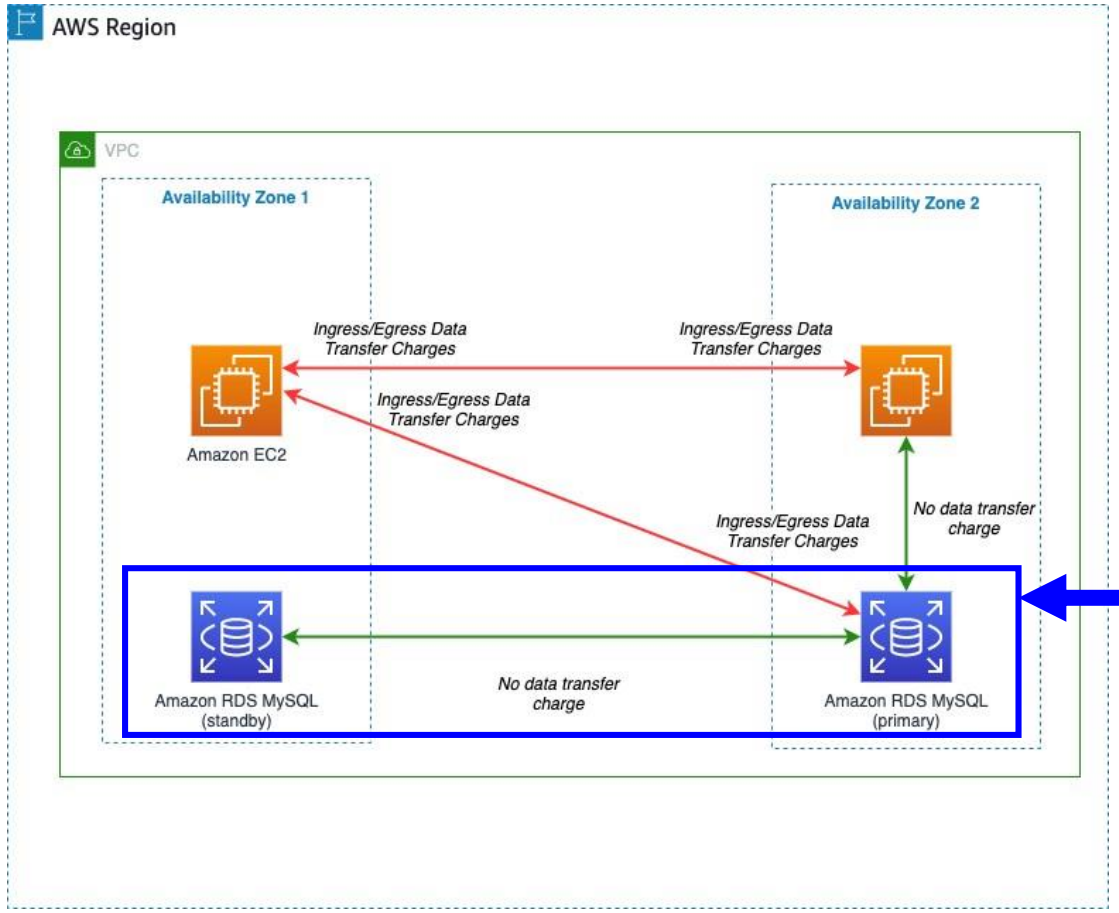
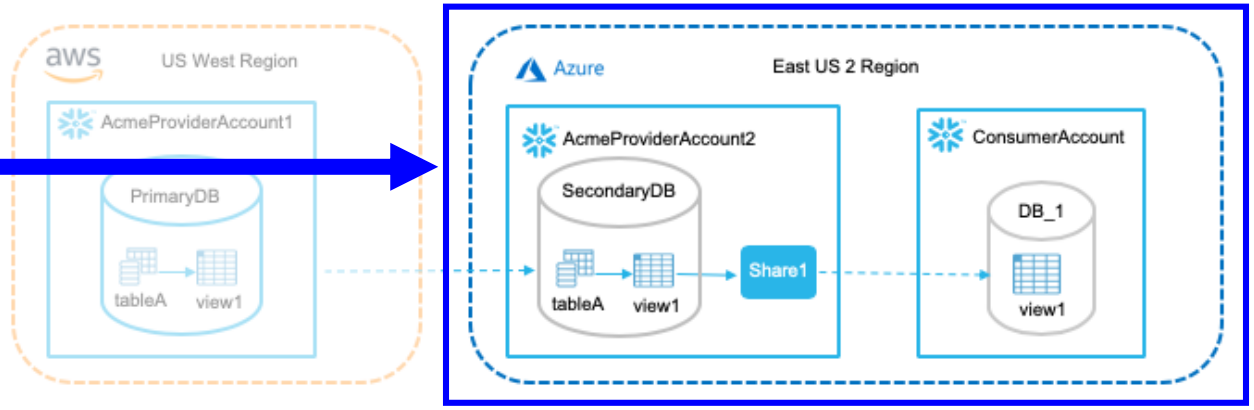


Figure 3. Workload components across Availability Zones

Availability Zone이 다름에도

No data transfer charge



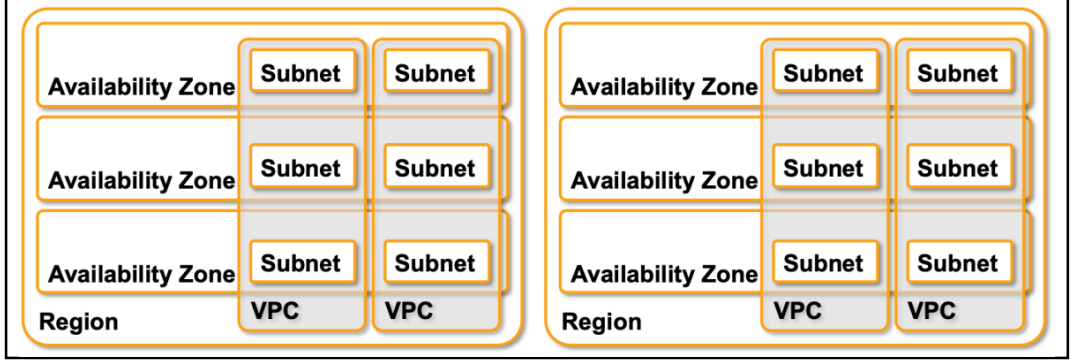
출처:
AWS, "Overview of Data Transfer Costs for Common Architectures", 2021 (accessed on 2022-01-25 https://aws.amazon.com/ko/blogs/architecture/overview-of-data-transfer-costs-for-common-architectures/)

CSP의 네트워크 형상 및 관계 분석

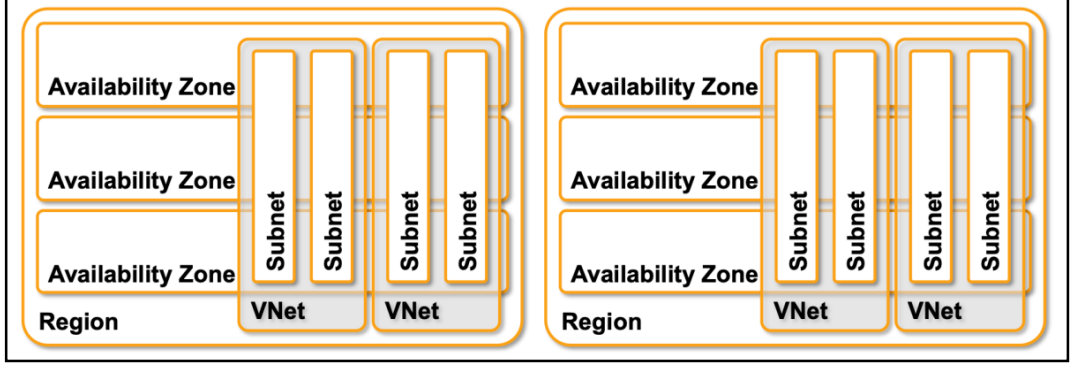
구역 구분 참고, 주관적임:
(같음) 물리적 구역 – Region, Availability Zone
(다름) 논리적 구역 – VPC/vNet/VPC network, Subnet/vSwitch

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

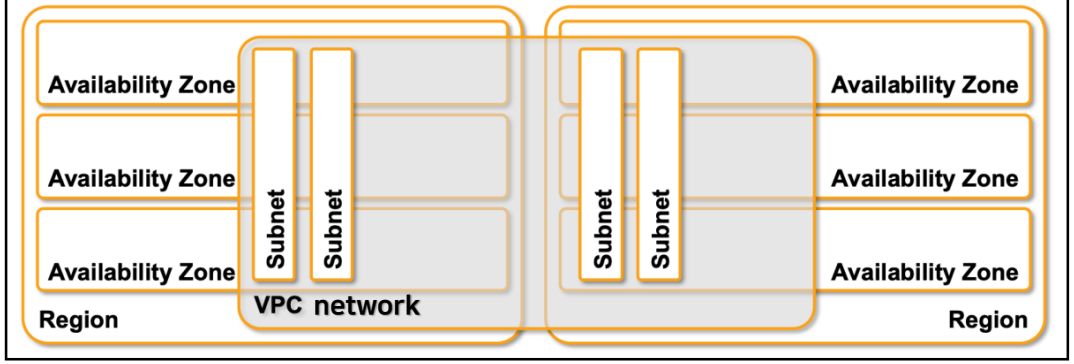
Amazon Web Services (AWS)



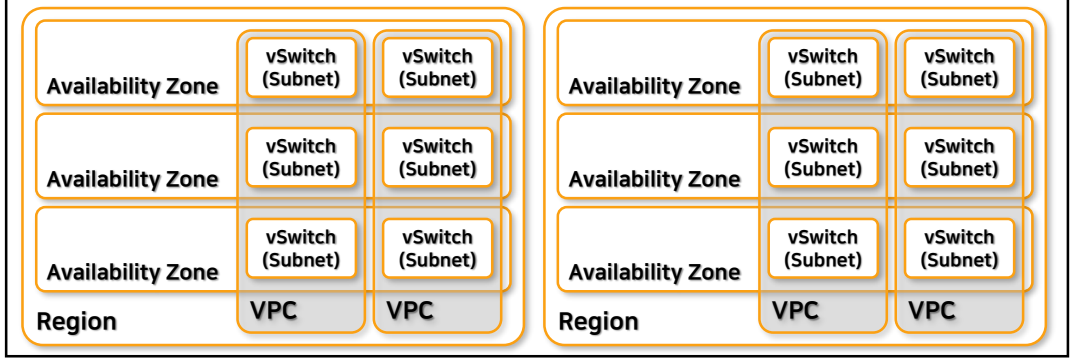
Microsoft Azure (MS Azure)



Google Cloud Platform (GCP)



Alibaba Cloud / IBM Cloud

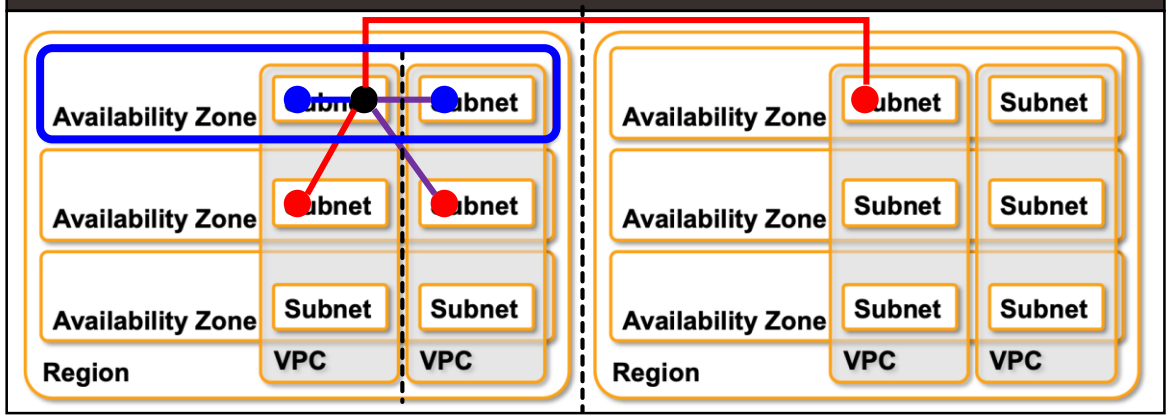


출처:
ipSpace.net, "Virtual Networks and Subnets in AWS, Azure, and GCP", 2021 (accessed on 2022-01-25 https://blog.ipspace.net/2021/02/vpc-subnets-aws-azure-gcp.html)
ipSpace.net, "Availability Zones and Regions in AWS, Azure and GCP", 2021 (accessed on 2022-01-25 https://blog.ipspace.net/2021/02/public-cloud-regions-availability-zones.html)
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)

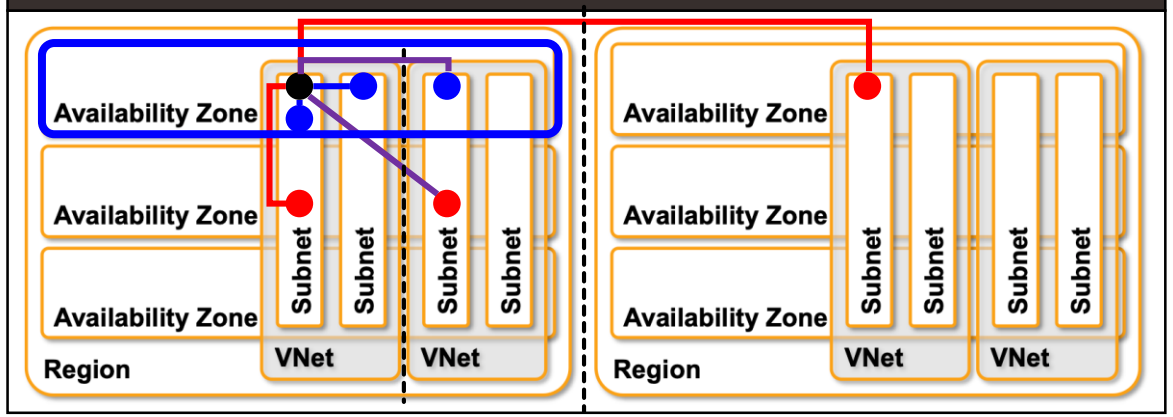
Data transfer billing 고찰

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

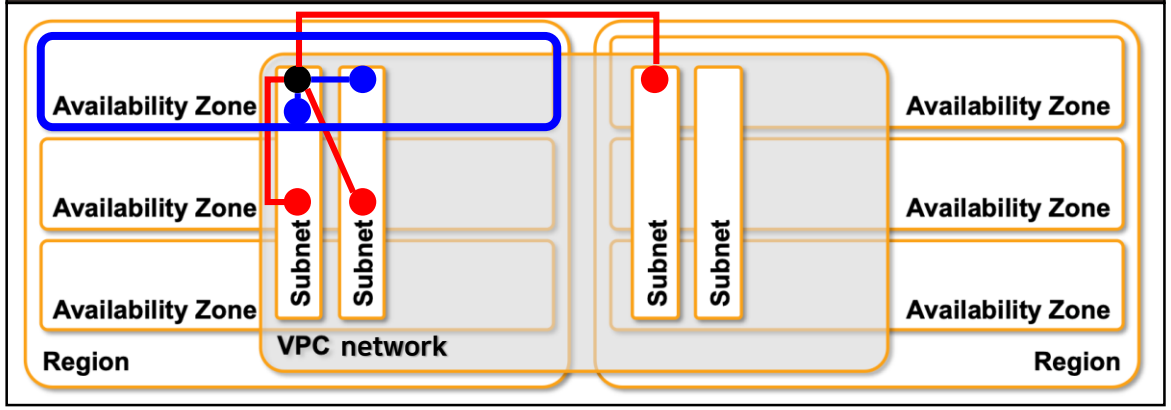
Amazon Web Services (AWS)



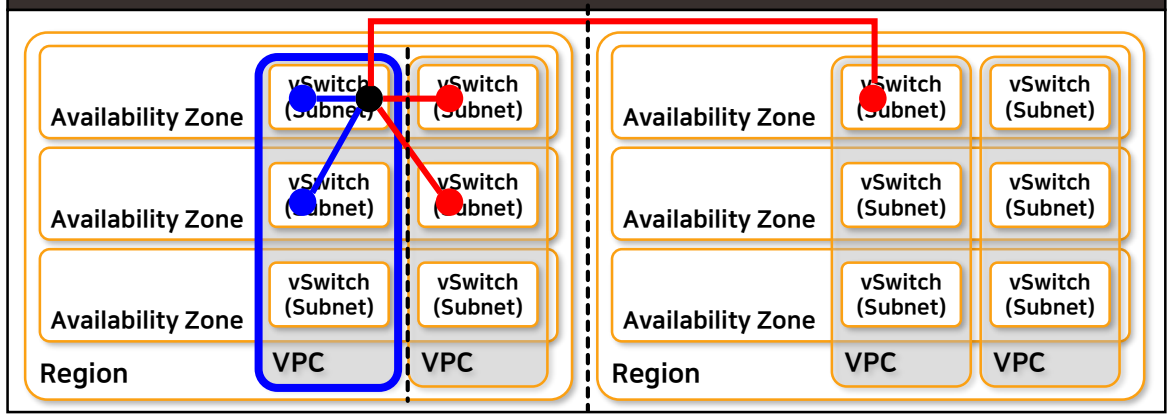
Microsoft Azure (MS Azure)



Google Cloud Platform (GCP) – 전역 설정 (지역 설정의 경우는 MS Azure와 동일)

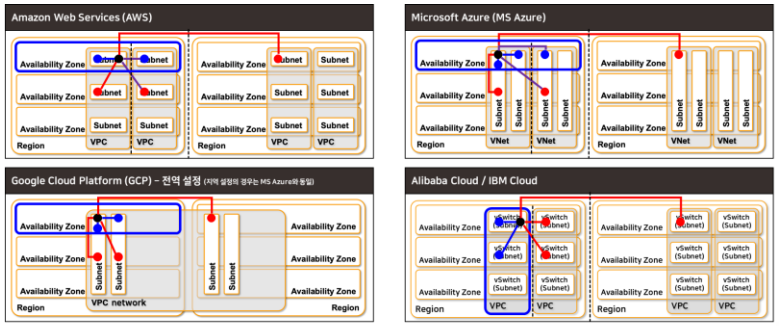


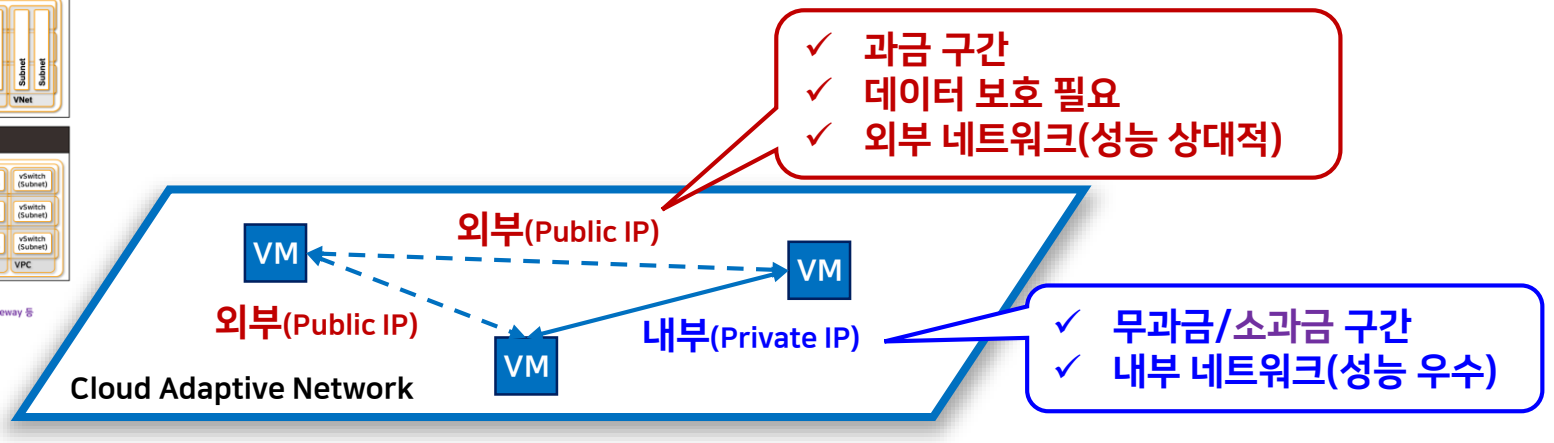
Alibaba Cloud / IBM Cloud



[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

비용에 적응하는 Cloud Adaptive Network





CB-Larva

클라우드바리스타 실험실

CB-Tumblebug

멀티클라우드 인프라 서비스 통합 운용/관리

CB-Spider

멀티클라우드 인프라 연동

Billing & performance

비용을 고려한 Cloud Adaptive Network

✓ 멀티클라우드 가상/공통 네트워크

클라우드 정보

✓ 멀티클라우드 인프라
✓ 글로벌 스케일 인프라

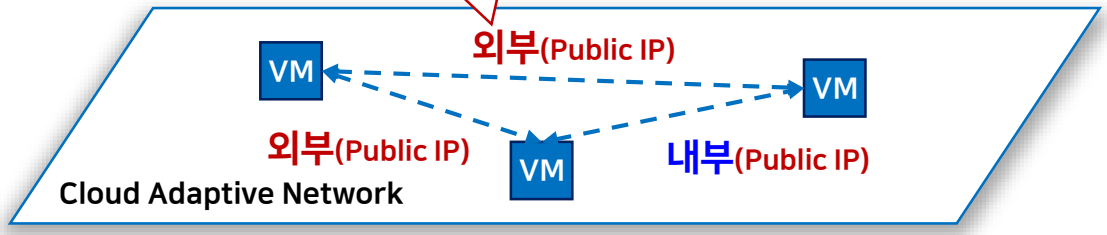
AWS

Google Cloud

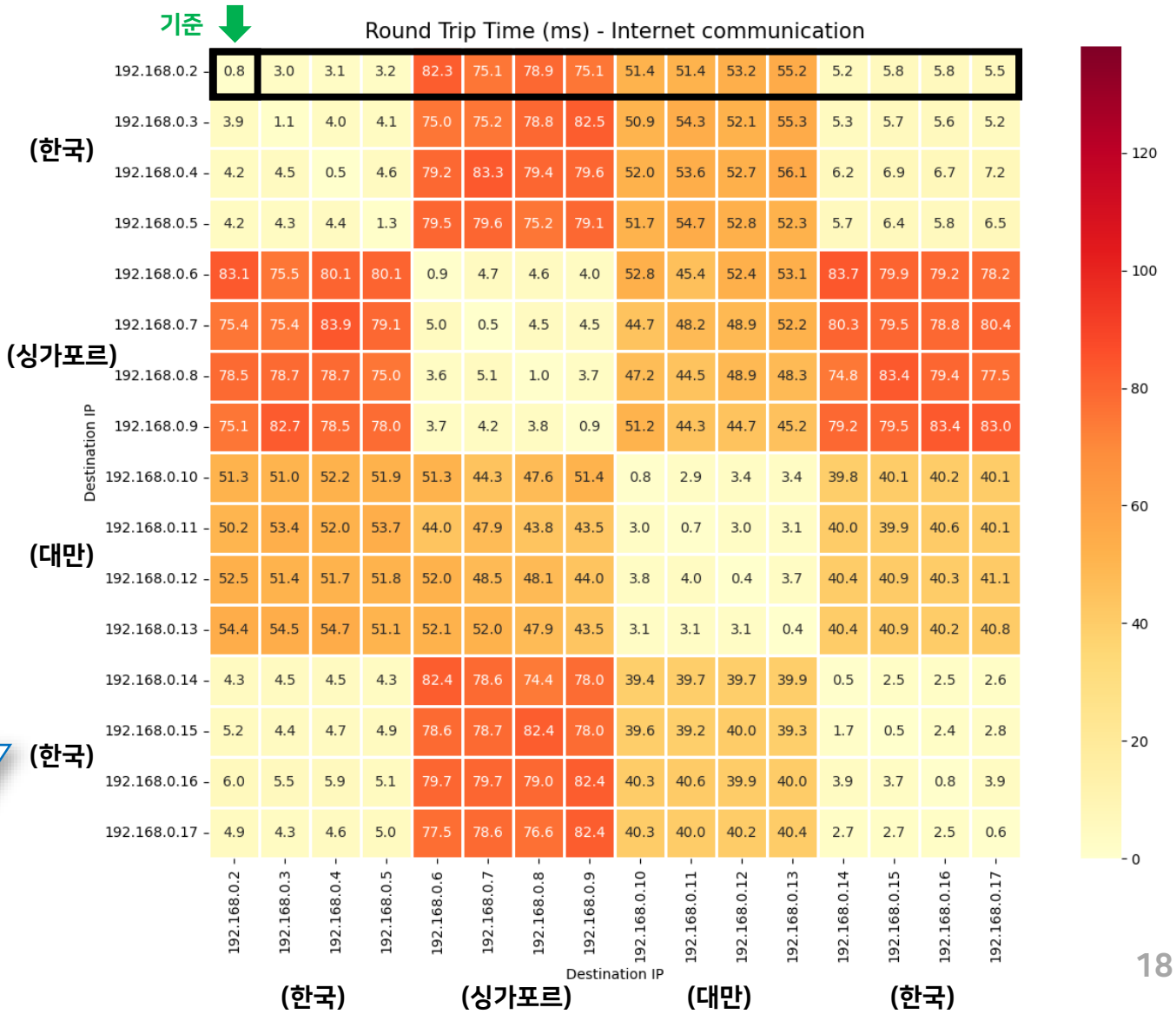
Microsoft Azure

성능 실험 1: 인터넷 통신(Public IP)

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



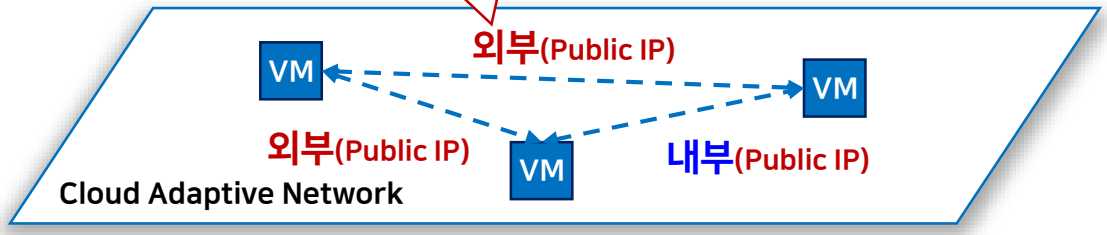
* Network tunneling



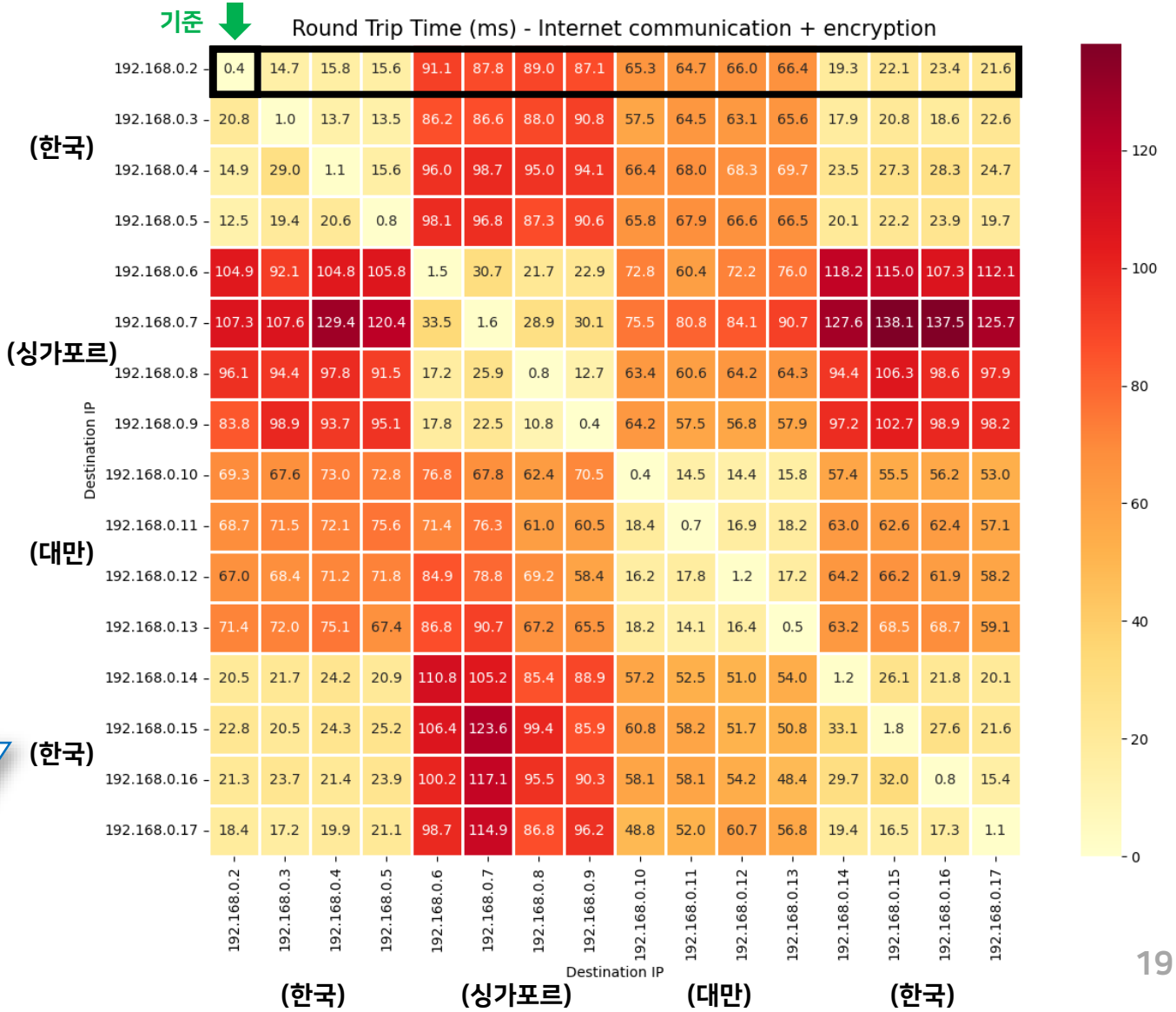
성능 실험 2: 인터넷 통신 + End-to-end encryption

$
(Compute)

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



* Network tunneling

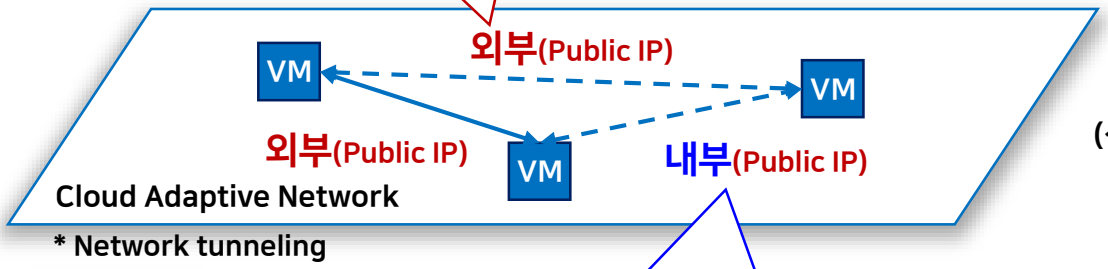


성능 실험 3: 비용을 고려한 통신 + End-to-end encryption

#Private IP/Public IP 선택 #외부통신암호화

$ (Compute)

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



* Network tunneling

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

절감! ~~$~~ (Compute)

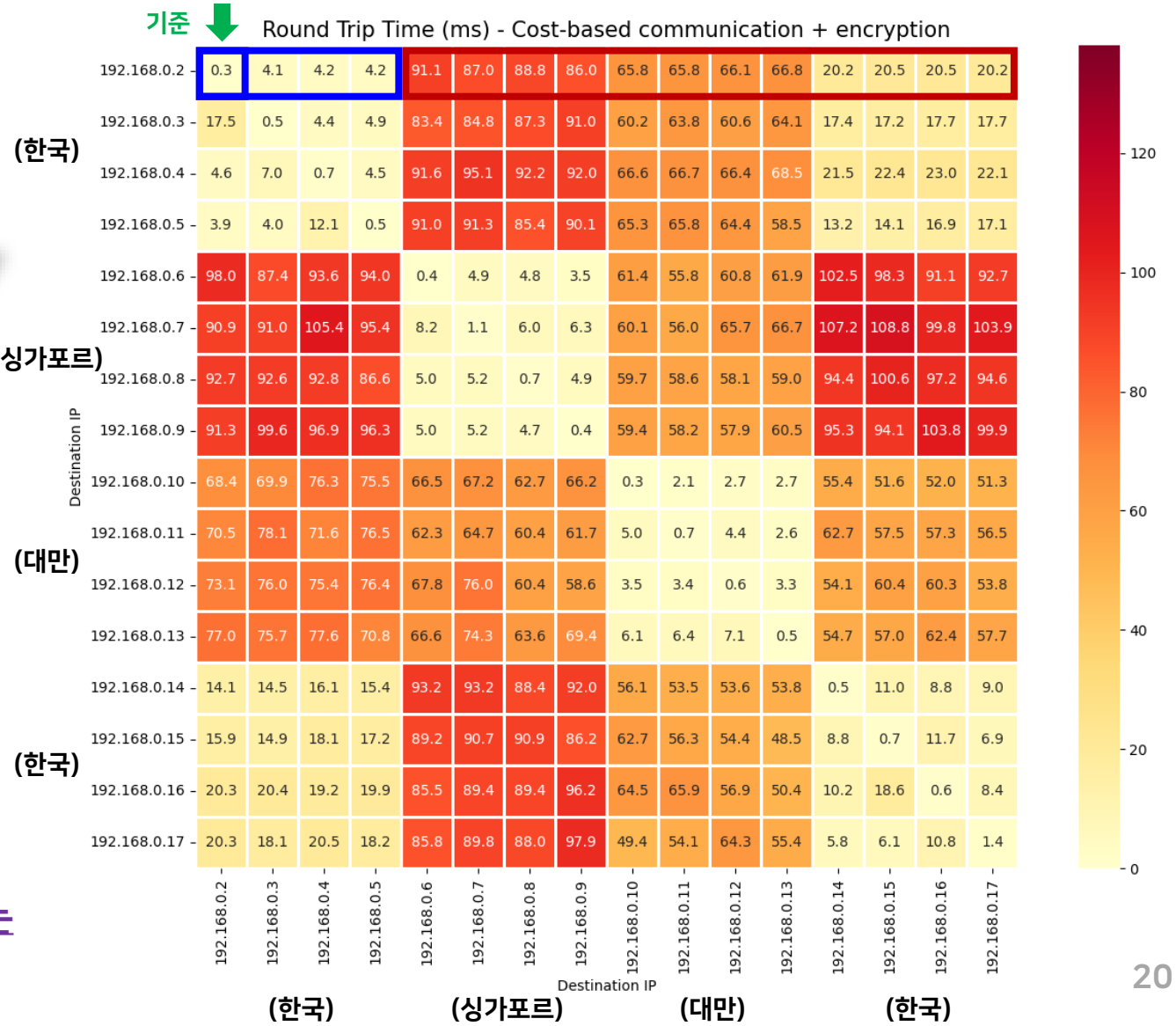
절감! ~~$~~ (Network)

Q. 두 서비스 간에 상호작용이 많다면 어떻게 배치 하시겠습니까?

과금 구간 통신을 고려 및 최적화하여

1) 인프라 구성/서비스 배치가 요구됨, 또는

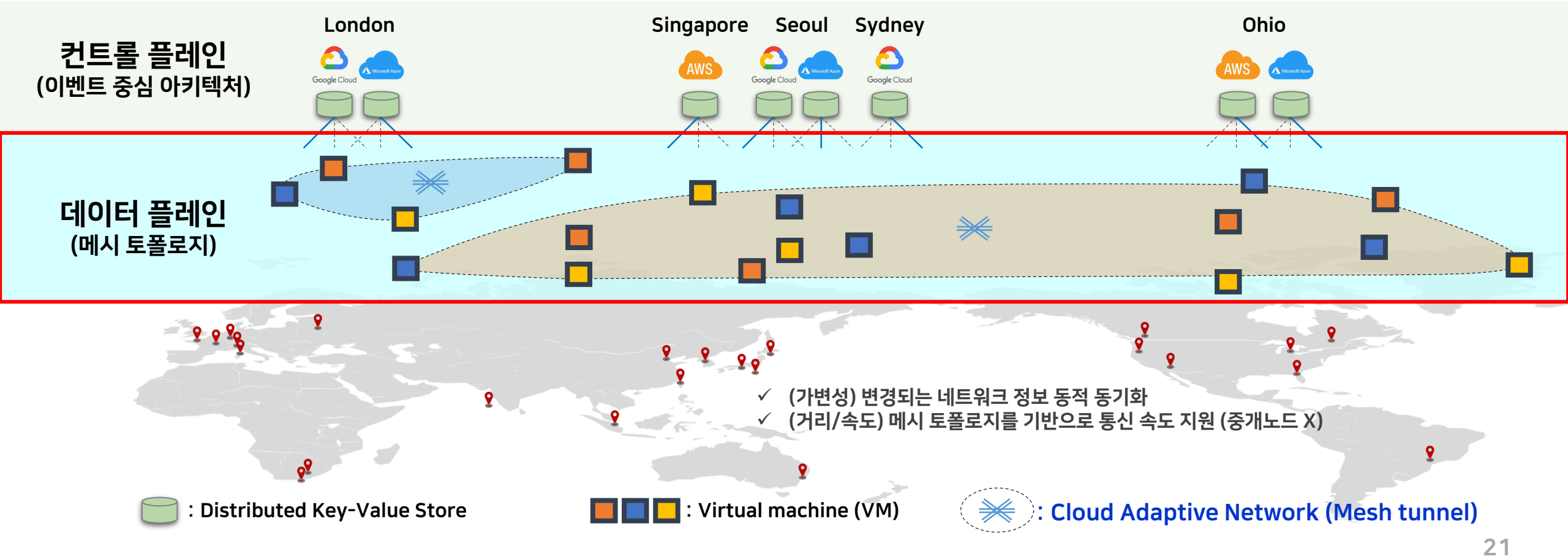
2) 다른 시각의 아이디어가 필요함



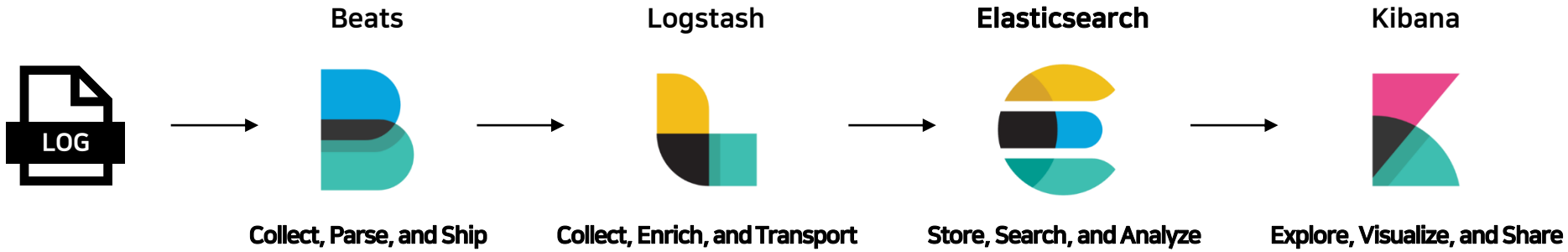
도전과제(Challenge)

#멀티클라우드 #분산시스템 #디버깅 #로깅

멀티클라우드에서 수십, 수백대의 VM 상에 배치되는 cb-network agent의 디버깅이 어려움



분산 로깅을 위한 도구 도입



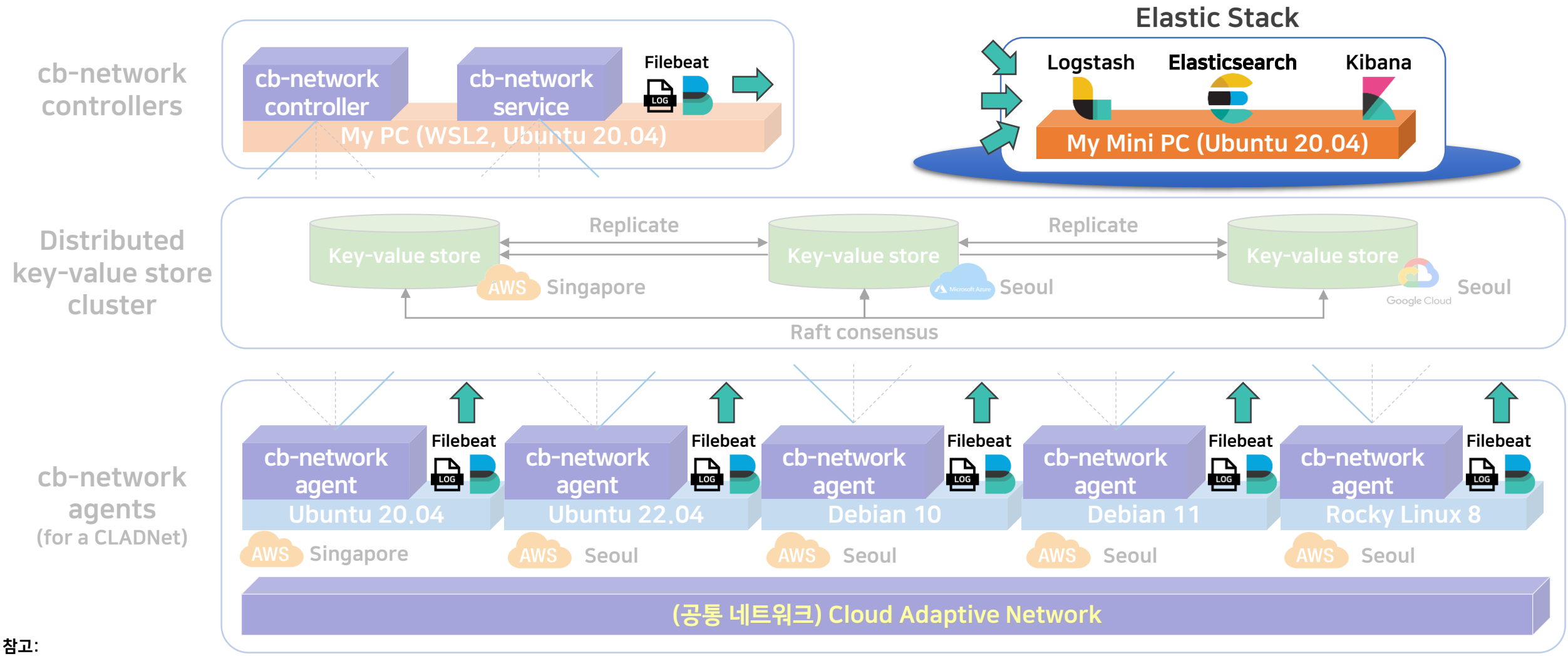
Elastic Stack의 로그 수집 컴포넌트 및 흐름

- 선정 기준
  - 가능한 빠르게 적용해 볼 수 있는 도구
  - 오픈소스, 비상용 프로젝트에 무료, cb-network system과 느슨한 결합, 데이터 전처리 가능, 시각화/분석 도구(웹), 등

- Elastic Stack 개요
  - Elasticsearch: Store, Search, and Analyze
  - Kibana: Explore, Visualize, and Share
  - Logstash: Collect, Enrich, and Transport
  - Beats: Collect, Parse, and Ship

분산 디버깅에 적응하는 Cloud Adaptive Network

✓ 각 cb-network component가 파일에 로그 기록 (← Loosely coupled →) ✓ Filebeat가 Log 파일을 읽어 Logstash로 전송



참고:
Distributed logging by Elastic Stack 적용기, https://github.com/cloud-barista/cb-coffeehouse/blob/main/docs/Distributed-logging-by-Elastic-Stack.md

로깅 개선: Details

cb-network 시스템 전반의 로깅 개선

✓ 각 cb-network component에 service ID 부여

cb-network controller | cb-network service | cb-network admin-web | cb-network agent

(As-Is)

controller-[ID_hash] service-[ID_hash] admin-web-[ID_hash] agent-[ID_hash]

service_id	log_lev...	function	log_message
controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:122, main.watchHostNetworkInformation()	Watch with prefix - /regis
controller-cclgkuatiahsik0ck4d0	INFO	controller.go:462, main.main()	Waiting for all goroutines
controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:114, main.watchHostNetworkInformation()	Start.....
controller-cclgkuatiahsik0ck4d0	INFO	controller.go:453, main.main()	The etcdClient is connecte
controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:107, main.init.0()	Load /mnt/d/Dev/Cloud-Bari
controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:431, main.main()	Start.....

✓ Log message에 세부 정보 포함

시작, 수행 내용, 끝, 수행시간 기록

log_message
Acquire a lock
Lock acquired for '/registry/cloud-adaptive-network/distributed-lock/peer/clad00perf'
Release a lock
Lock released for '/registry/cloud-adaptive-network/distributed-lock/peer/clad00perf'
Elapsed time for locking (sec): 0.203

etcd 입/출력 관련 key-value 크기, 개수 출력

TransactionRequest size (bytes): total_size: 461, networking_rule_count: 3
GetResponse size (bytes): total_size: 493, header_size: 27, kvs_size: 466, kvs_count: 1
TransactionRequest size (bytes): total_size: 462, networking_rule_count: 3
GetResponse size (bytes): total_size: 492, header_size: 27, kvs_size: 465, kvs_count: 1
GetResponse size (bytes): total_size: 249, header_size: 27, kvs_size: 222, kvs_count: 1
GetResponse size (bytes): total_size: 1946, header_size: 28, kvs_size: 1918, kvs_count: 4
PutRequest size (bytes): total_size: 361
PutRequest size (bytes): total_size: 363
GetResponse size (bytes): total_size: 27, header_size: 27, kvs_size: 0, kvs_count: 3

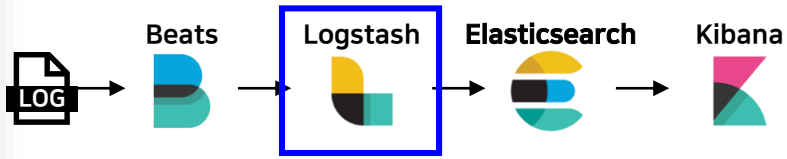
로깅 개선: Patterns

Log message 패턴화 by Logstash

- ✓ Log message의 타임 스탬프 추출 및 활용 (Elastic Slack의 timestamp로는 분산 시스템 동작순서를 구분할 수 없었음)
- ✓ Kibana interface에서 Log message 세부 정보의 필드화

Columns 1 field sorted

@timestamp	message.keyword
Aug 17, 2022 @ 21:44:04.751	[cb-network].[ERROR]: 2022-08-17 21:44:04 admin-web.go:183, main.WebsocketHandler() - websocket: close 1001 (going away)
Aug 17, 2022 @ 21:44:04.363	[cb-network].[DEBUG]: 2022-08-17 12:44:00 cb-network.go:635, github.com/cloud-barista/cb-larva/poc-cb-net/pkg/cb-network.(*CBNetwork).CloseCBNetworkInterface() - End.....
Aug 17, 2022 @ 21:44:04.363	[cb-network].[DEBUG]: 2022-08-17 12:44:00 agent.go:375, main.updatePeerState() - Put "/registry/cloud-adaptive-network/peer/c8fifu3rn5bc06nth0qg/cbudo979cjs31g976k40"
Aug 17, 2022 @ 21:44:04.363	[cb-network].[DEBUG]: 2022-08-17 12:44:00 agent.go:368, main.updatePeerState() - Start.....
Aug 17, 2022 @ 21:44:04.363	[cb-network].[DEBUG]: 2022-08-17 12:44:00 agent.go:382, main.updatePeerState() - End.....
Aug 17, 2022 @ 21:44:04.363	[cb-network].[DEBUG]: 2022-08-17 12:44:01 agent.go:882, main.main() - End cb-network agent .....
Aug 17, 2022 @ 21:44:04.362	[cb-network].[DEBUG]: 2022-08-17 12:44:00 cb-network.go:629, github.com/cloud-barista/cb-larva/poc-cb-net/pkg/cb-network.(*CBNetwork).CloseCBNetworkInterface() -



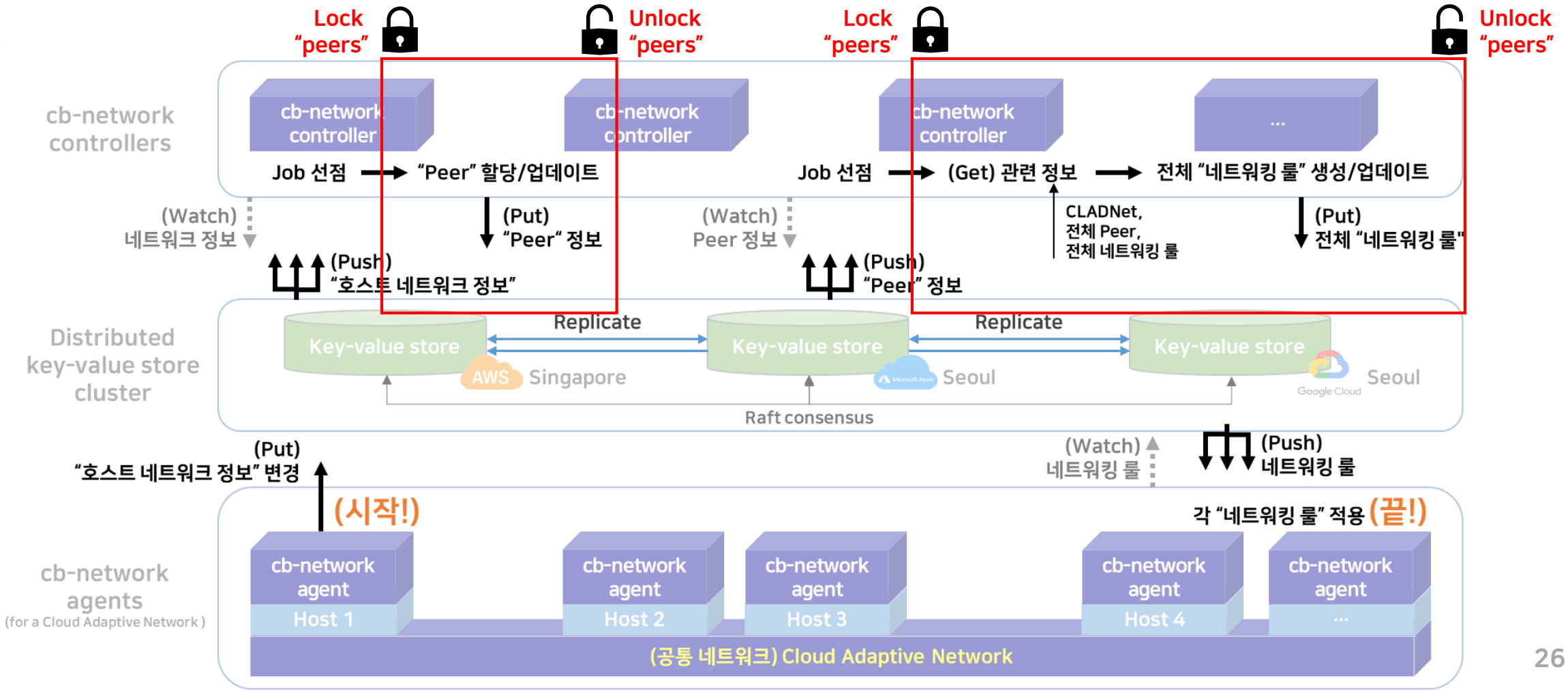
Columns 2 fields sorted

@timestamp	log_timestamp	log_time_string	service_id	log_lev...	function	log_message
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.470	2022-09-21 21:58:33.4706	controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:122, main.watchHostNetworkInformation()	Watch with prefix - /registry/cloud-adaptive-network/host-network-information
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.401	2022-09-21 21:58:33.4017	controller-cclgkuatiahsik0ck4d0	INFO	controller.go:462, main.main()	Waiting for all goroutines to finish
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.401	2022-09-21 21:58:33.4017	controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:114, main.watchHostNetworkInformation()	Start.....
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.401	2022-09-21 21:58:33.4017	controller-cclgkuatiahsik0ck4d0	INFO	controller.go:453, main.main()	The etcdClient is connected.
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.400	2022-09-21 21:58:33.4002	controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:107, main.init.0()	Load /mnt/d/Dev/Cloud-Barista/cb-larva/poc-cb-net/config/log_conf.yaml
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.400	2022-09-21 21:58:33.4002	controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:431, main.main()	Start.....
2022-09-21 21:58:33.7670	2022-09-21 21:58:33.393	2022-09-21 21:58:33.3930	controller-cclgkuatiahsik0ck4d0	DEBUG	controller.go:106, main.init.0()	Load /mnt/d/Dev/Cloud-Barista/cb-larva/poc-cb-net/config/config.yaml

참고:
Transforming logs with Logstash 적용기, https://github.com/cloud-barista/cb-coffeehouse/blob/main/docs/Transforming-logs-by-Logstash.md

(Challenge) 가상 네트워크 정보 동적 업데이트시 성능

소스코드/데이터구조 레벨의 분산 시스템 전반을 분석 ➔ 이슈: 1) 빈번한 etcd I/O, 2) 분산 락, 3) 높은 복잡도

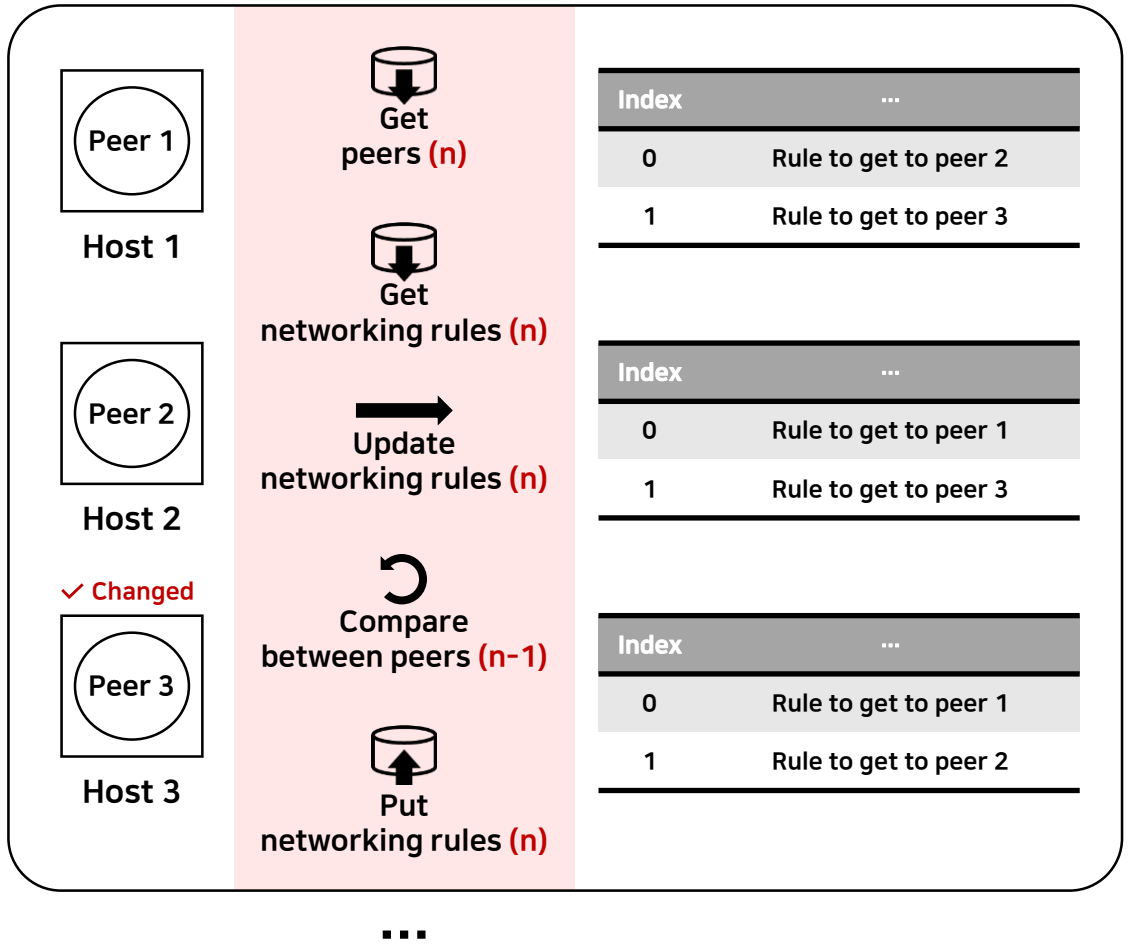


cb-network controller에서 네트워킹 룰 업데이트 방법

Issue: 높은 복잡도

cb-network controller

* RuleType: Cost-prioritized communication

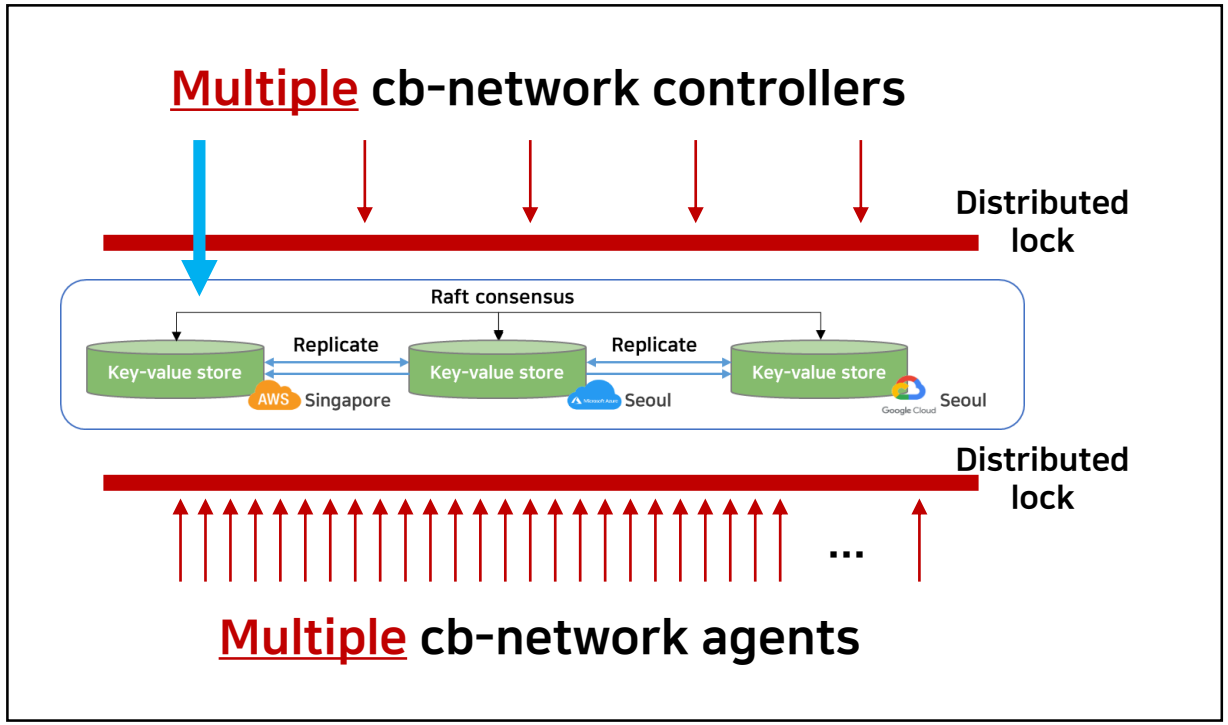


...

Issue: 빈번한 분산 락

Critical section !

key: peers
key: networking rules

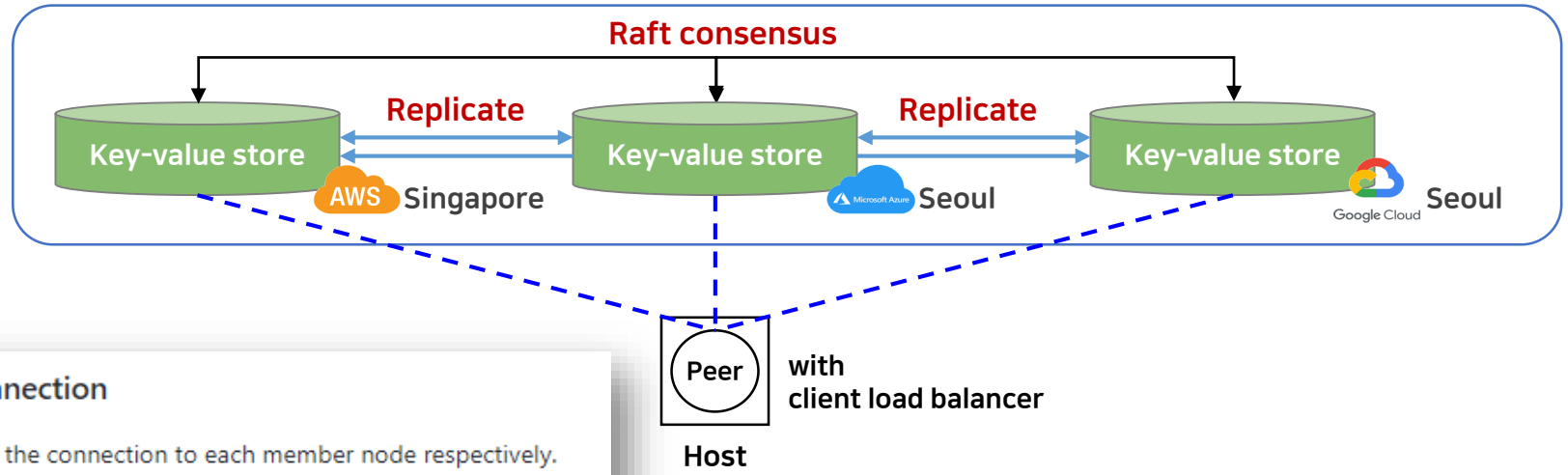


➔ 이렇다 하더라도… 이 정도를 수행하면서 느리다고 체감되면 안될 것 같은데 말이죠……

#진짜 원인

(Cause) Pretty long latency from etcd cluster

Distributed key-value store cluster



Case 1: individual connection

A test program established the connection to each member node respectively.
I was able to get a response to the Status() call right away, but not for other calls.
I think it takes time to perform Raft (consensus algorithm) in a cluster.
As a result, it has an average response time.

Test result

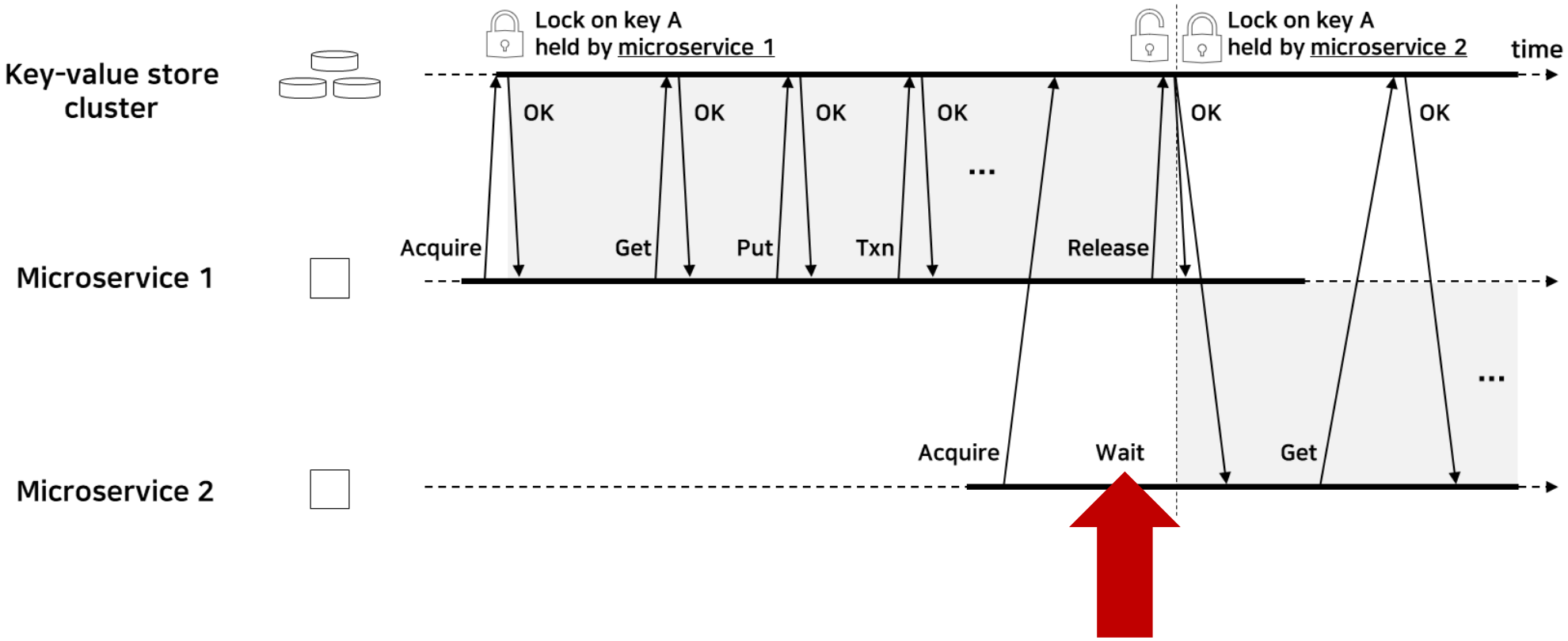
Test case	Member 1	Member 2	Member 3
Status() call (like ping)	232ms	18ms	15ms
Health-check REST call	154ms	172ms	159ms
Get() call	149ms	169ms	154ms
Put() call	169ms	175ms	154ms

System people 관점 ➔ 꽤 긴 지연시간

Multi-cloud people 관점 ➔ 염두에 두어야할 지연시간

Source: https://github.com/cloud-barista/cb-larva/issues/315

(Cause) Wait time by distributed-lock for concurrency



운용 효율에 적응하는 Cloud Adaptive Network

Idea?!

1) agent가 정보를 일부 보유

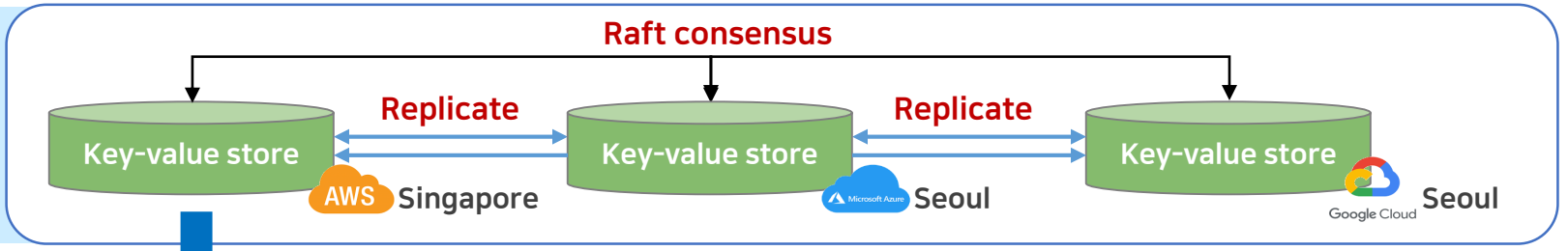
2) Key-value에 적합한 자료 구조

운용 효율에 적응하는 Cloud Adaptive Network

이슈: 1) 빈번한 etcd I/O , 2) 분산 락, 3) 높은 복잡도

How I've resolved it? 최초 확실한 동기화 ➔ 분산-락 최소화 ➔ 동작 매커니즘 단순화

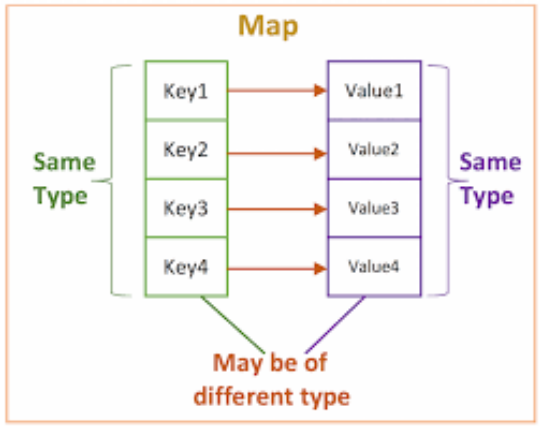
- ✓ agent가 정보를 일부 보유
- ✓ 필요한 정보만 최초 확실한 동기화,
- ✓ 그룹 단위 분산-락 (이후, 개별 단위 분산-락)

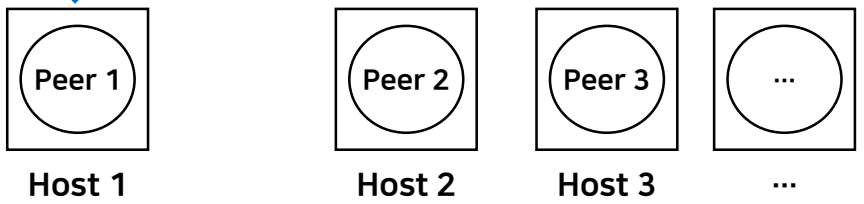


On initializing stage, synchronize all peers in local (memory)

1. Lock peers
2. Synchronize peers (n)
3. Unlock peers

A data structure:
Map (e.g., HashMap)





Agent가 정보를 일부 보유

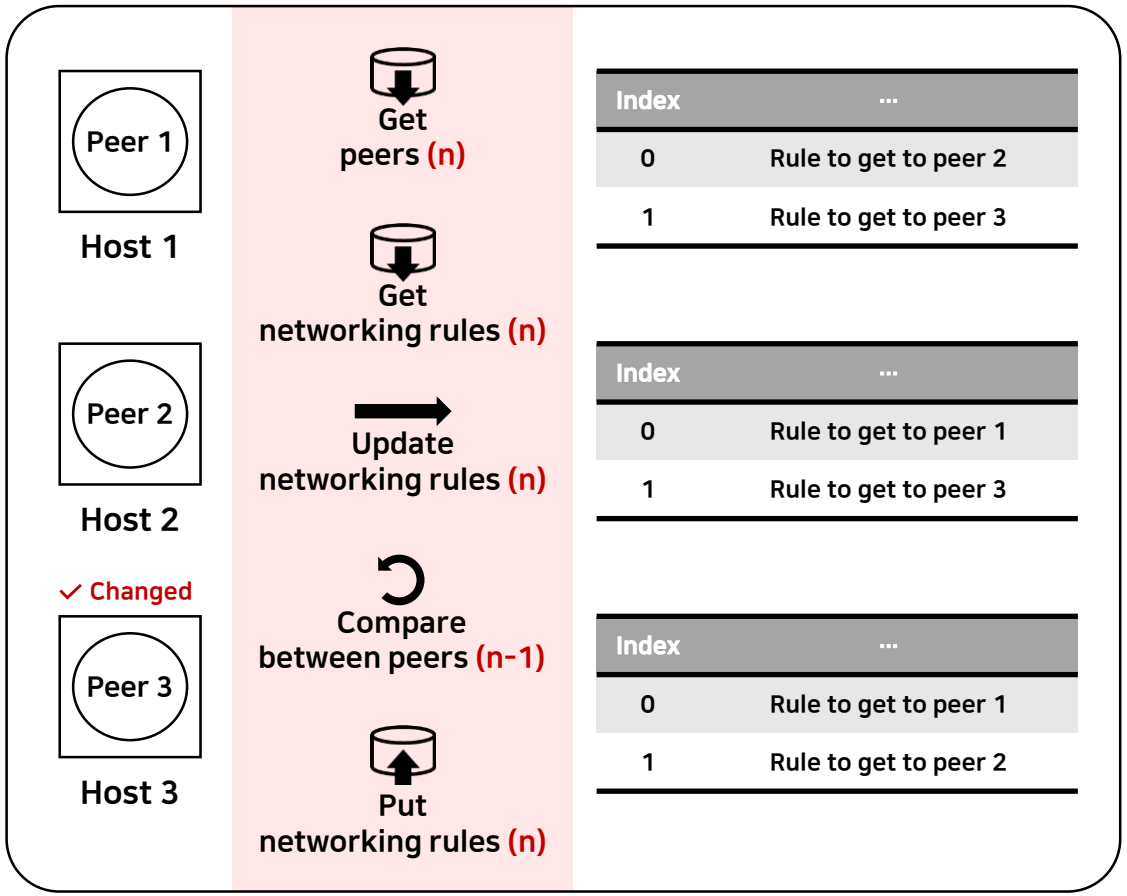
참고

Q) controller에서도 할 수 있지 않을까요?
A) 이슈가 있어요 ^^;;
(동기화 이슈) multi-controller간 데이터 처리 필요.
- 임계 구역, Latency, 등
(방대한 정보량 이슈) 메모리에서 여러 CLADNet 정보 관리 필요.
- 각 Networking Rule 정보, 각 peer 정보, 등

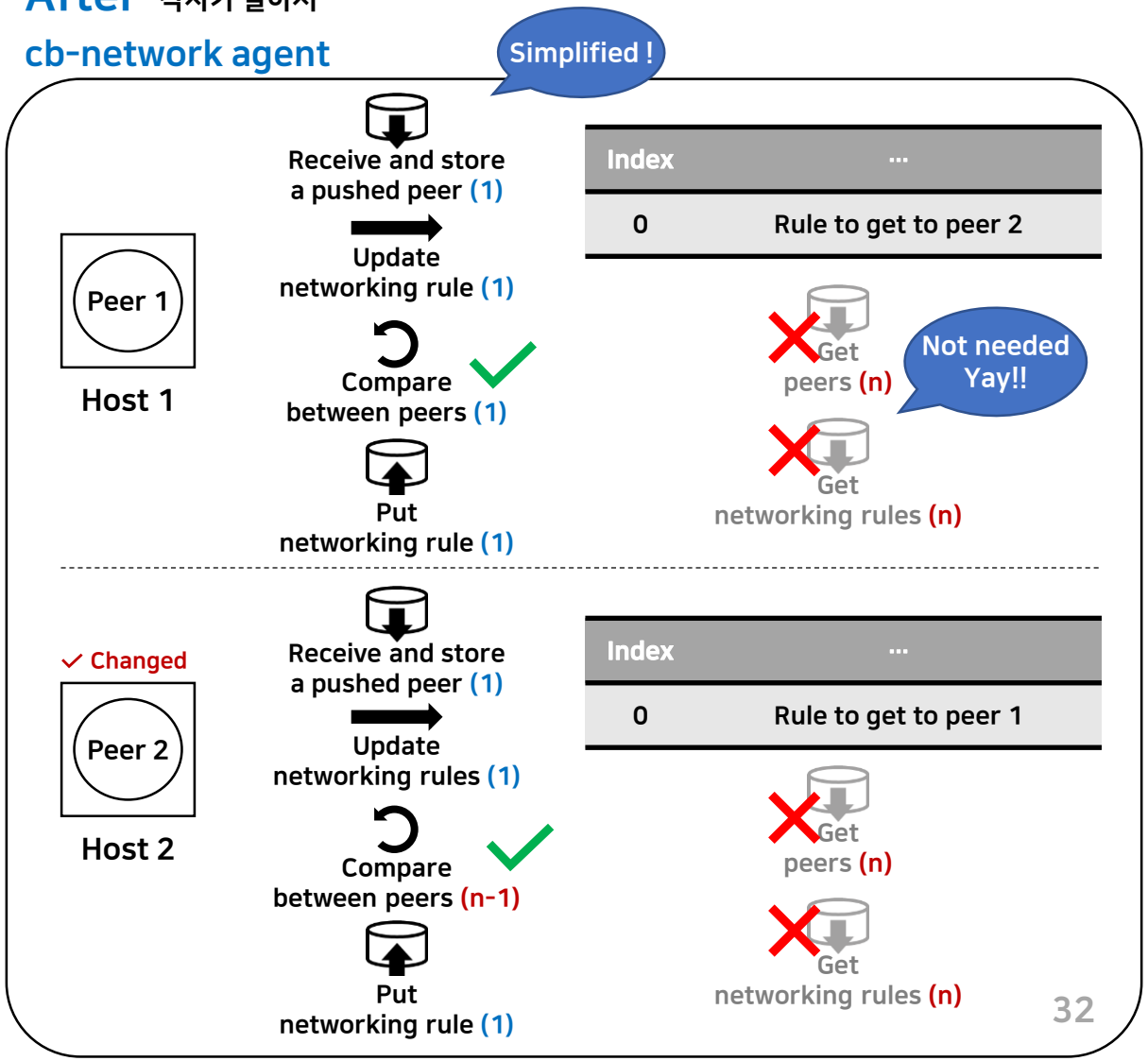
How I've resolved it: 워크플로 단순화 및 임계 구역 최소화

다 해주던 것을 **Before** → **After** 각자가 알아서

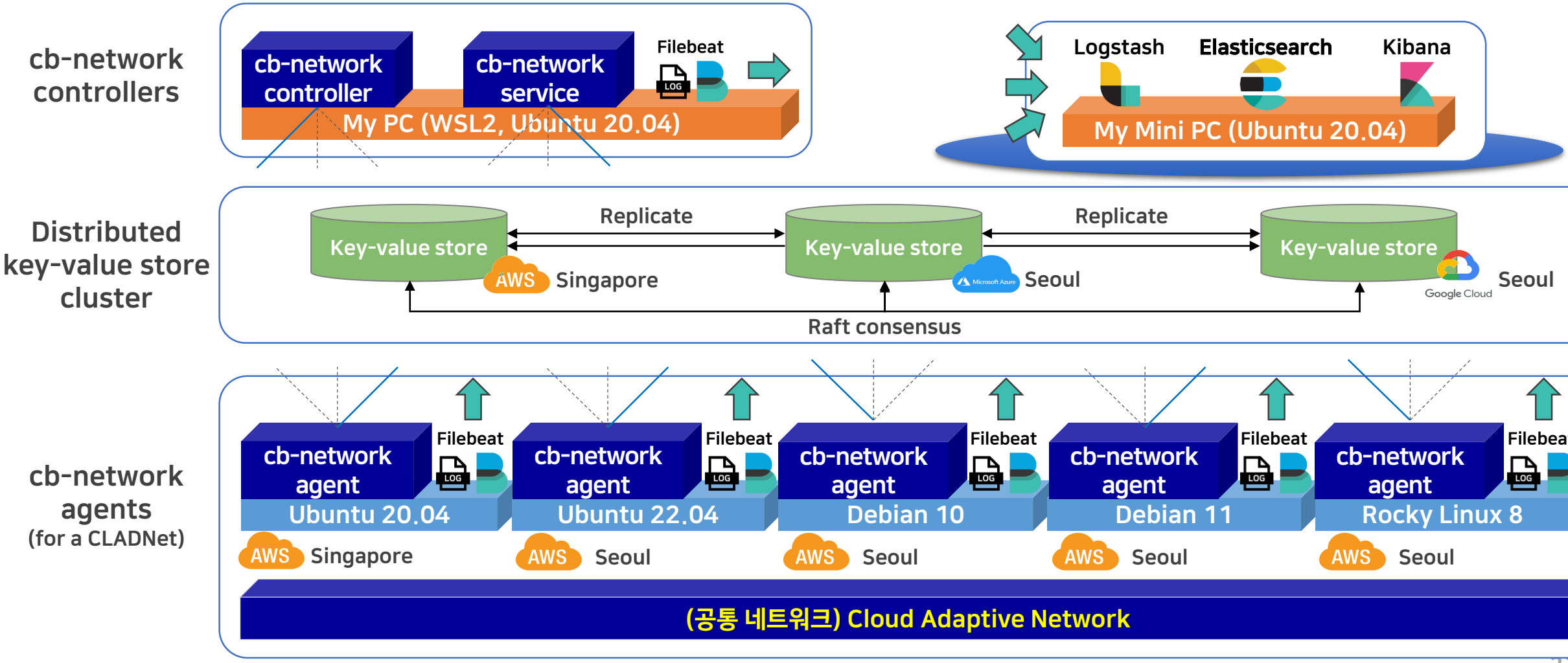
cb-network controller



cb-network agent



실험 환경: 분산 로깅이 적용된 cb-network 시스템



실험 환경: MCIS

MCIS Configuration

- NameSpace ID : ns01
- MCIS ID : clad00perf
- Number of Total VMs : 10
  - [1] VMs(5) = aws(5) * Replica(1)
  - [2] VMs(5) = gcp(5) * Replica(1)

실험 환경: 2 CSP, 5 Regions
각 리전의 VM 수를 1 부터 5까지 늘림
예) 10 VMs = 2 CSP * 5 Regions * 1 VM

Enabled Clouds and Regions

- [1] Cloud : aws (enabled regions : 5)
  [1,1] Region : aws-ap-northeast-2 (Asia Pacific (Seoul))
  - VM SPEC : t2.micro
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (ami-00379ec40a3e30f87)

  [1,2] Region : aws-ap-southeast-1 (Asia Pacific (Singapore))
  - VM SPEC : t3.medium
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (ami-061eb2b23f9f8839c)

  [1,3] Region : aws-ca-central-1 (Canada (Central))
  - VM SPEC : t2.micro
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (ami-0d0eaed20348a3389)

  [1,4] Region : aws-us-west-1 (US West (N. California))
  - VM SPEC : t2.micro
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (ami-0dd655843c87b6930)

  [1,5] Region : aws-us-east-1 (US East (N. Virginia))
  - VM SPEC : t2.micro
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (ami-085925f297f89fce1)

- [2] Cloud : gcp (enabled regions : 5)
  [2,1] Region : gcp-asia-northeast3 (Seoul South Korea [zone:a])
  - VM SPEC : e2-small
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/images/ubuntu-1804-bionic-v20220505)

  [2,2] Region : gcp-asia-east1 (Changhua County Taiwan [zone:a])
  - VM SPEC : e2-small
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/images/ubuntu-1804-bionic-v20220505)

  [2,3] Region : gcp-asia-east2 (Hong Kong [zone:a])
  - VM SPEC : e2-small
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/images/ubuntu-1804-bionic-v20220505)

  [2,4] Region : gcp-asia-northeast1 (Tokyo Japan [zone:a])
  - VM SPEC : e2-small
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/images/ubuntu-1804-bionic-v20220505)

  [2,5] Region : gcp-asia-northeast2 (Osaka Japan [zone:a])
  - VM SPEC : e2-small
  - VM DISK : default (default GB)
  - VM IMAGE : Ubuntu 18.04 (https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/images/ubuntu-1804-bionic-v20220505)

(참고) 실험 시나리오

목적: (성능 테스트) MCIS에 Cloud Adaptive Network 동시 설정 시, 성능 차이를 보임

- 초기 설정에 걸리는 시간
- 초기 설정 시 I/O 데이터 량

⬅ 임계 구역, 경쟁상태에서 실험 진행

1. Run CB-Spider container
2. Run CB-Tumblebug server
3. Initialize multi-cloud environment
4. Create an MCIS with x number of VMs

5. Run cb-network controller
6. Run cb-network service

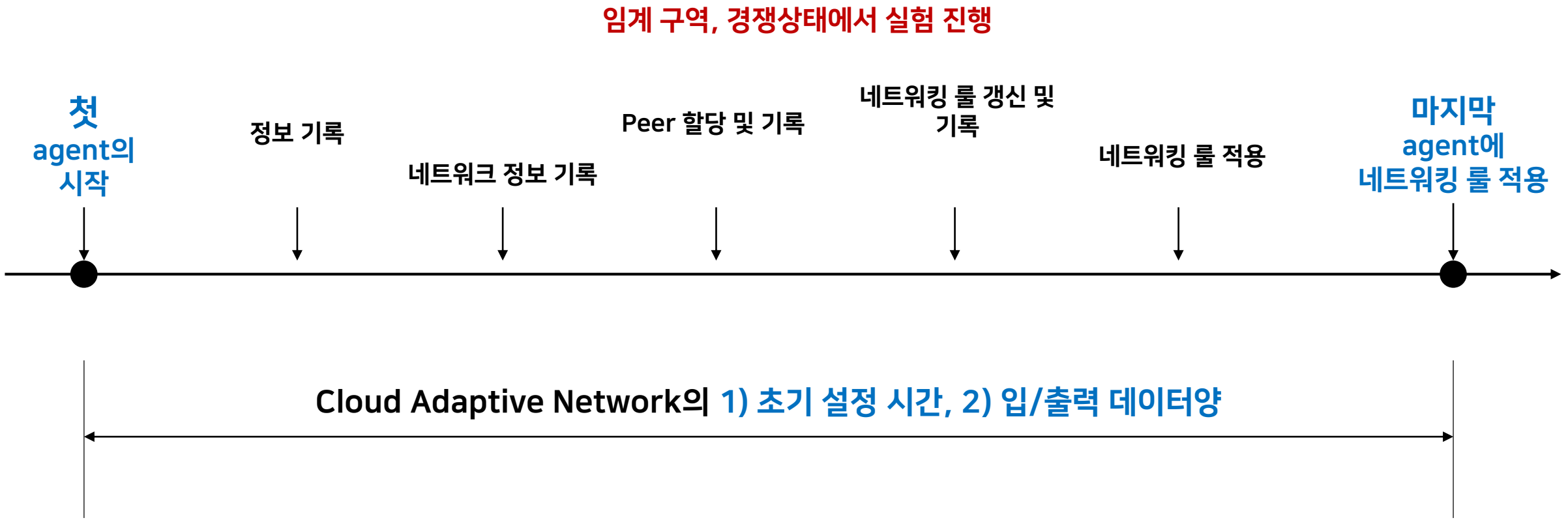
7. Open CB-Tumblebug Swagger UI
8. Deploy and run Filebeat on VMs (for distributed logging)
9. Deploy cb-network agent on VMs and configure Cloud Adaptive Network

10. Open Kibana interface
11. Filter logs and download csv of it

12. Clean the MCIS
13. Stop cb-network service, controller, CB-Tumblebug server, CB-Spider container

성능 측정 시기 및 항목

(n대의 VM에 Cloud Adaptive Network 설정)



(Output) 요약

운용 효율 개선 후

초기 설정 시간

평균 26% 감소

단위: Sec

# of VMs	Before	After	Rate
10	21.0	22.5	8%
20	40.5	30.4	-25%
30	61.7	42.7	-31%
40	102.4	59.0	-42%
50	137.2	85.9	-37%

입/출력 데이터양

평균 62% 감소

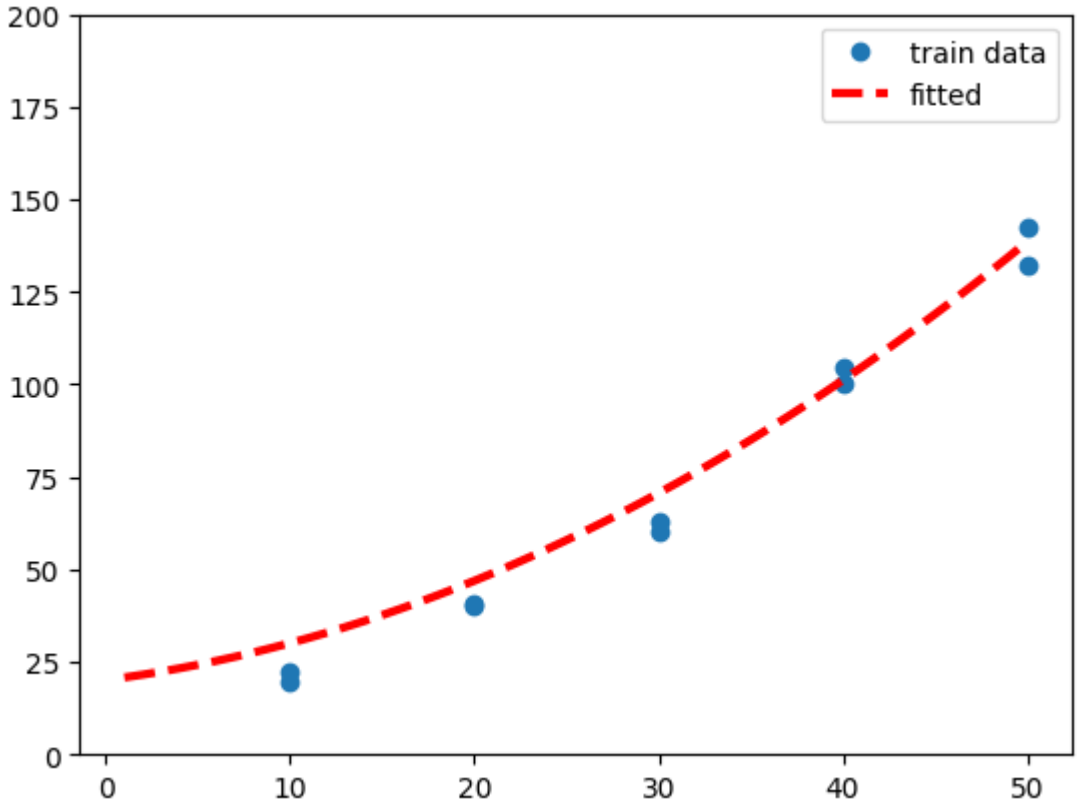
단위: KB

# of VMs	Before	After	Rate
10	419	145	-65%
20	2,781	862	-69%
30	7,046	2,608	-63%
40	16,592	5,838	-65%
50	21,302	11,021	-48%

(Output) 초기 설정 시간 - Cloud Adaptive Network 동시 설정 시

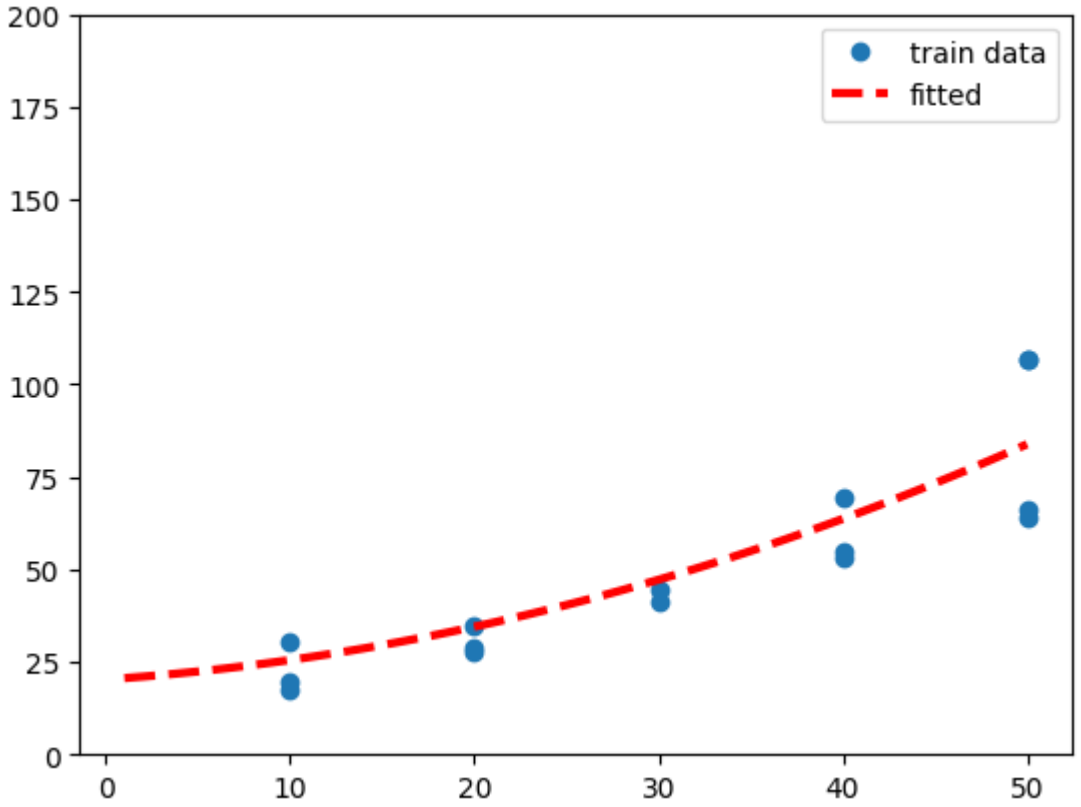
- 점: 실제 데이터 (VM이 10, 20, 30, 40, 50대 인 경우)
- 점선: Non-linear regression을 통해 도출한 추세선 ➔ VM 수에 따라 예측 가능

대조군(Before)



$h(x) = 0.6972x^2 + 0.2662x + 0.0098$

실험군(After)

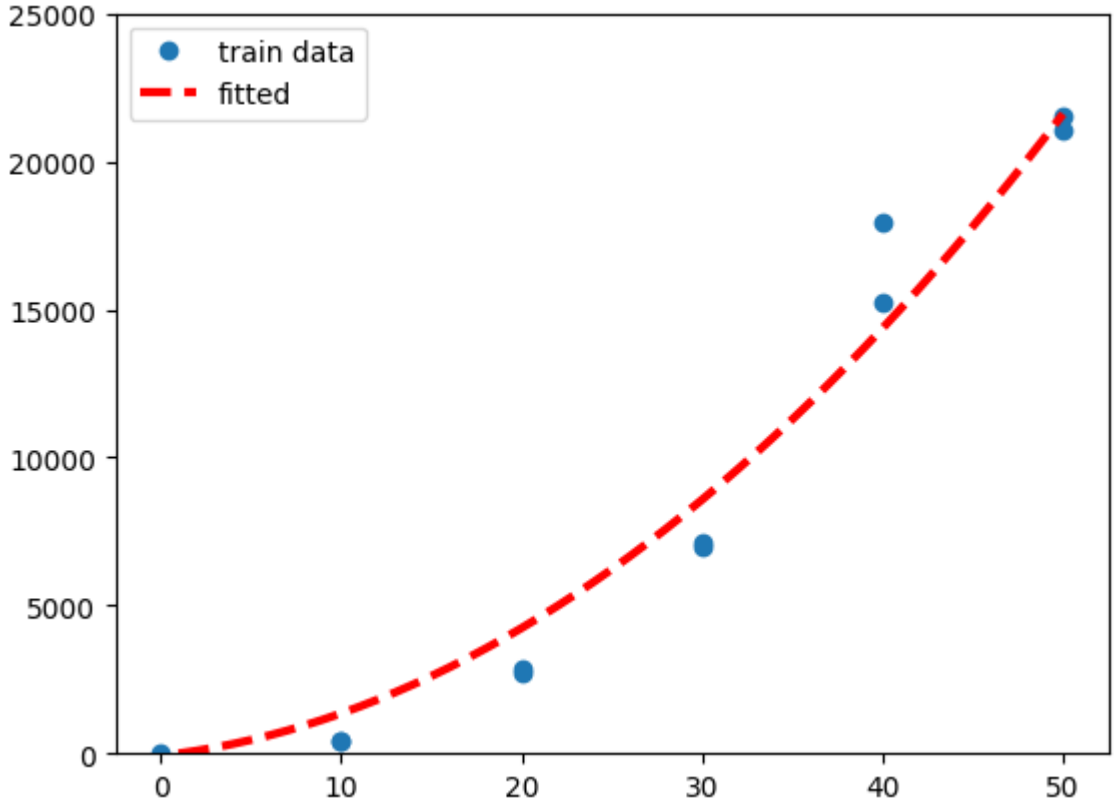


$h(x) = 0.5227x^2 + 0.1860x + 0.0329$

(Output) 입/출력 데이터양 - Cloud Adaptive Network 동시 설정 시

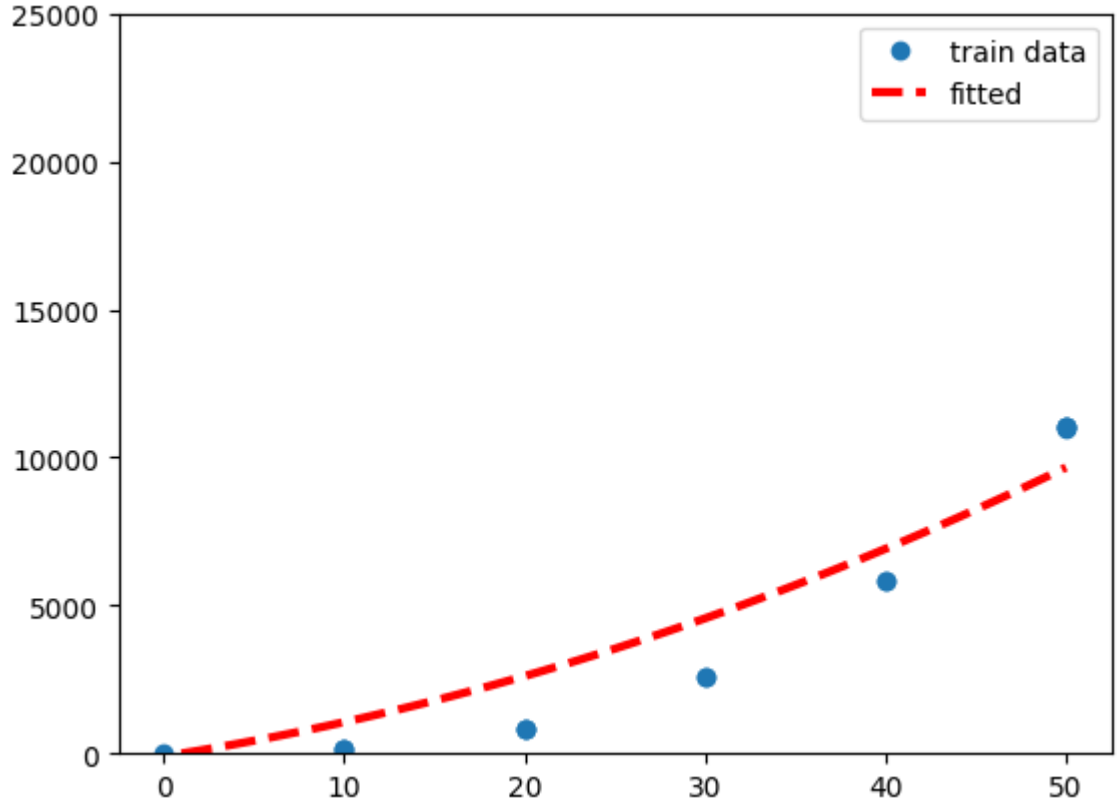
- 점: 실제 데이터 (VM이 10, 20, 30, 40, 50대 인 경우)
- 점선: Non-linear regression을 통해 도출한 추세선 ➔ VM 수에 따라 예측 가능

대조군(Before)



$h(x) = 0.8424x^2 + 0.1596x$

실험군(After)

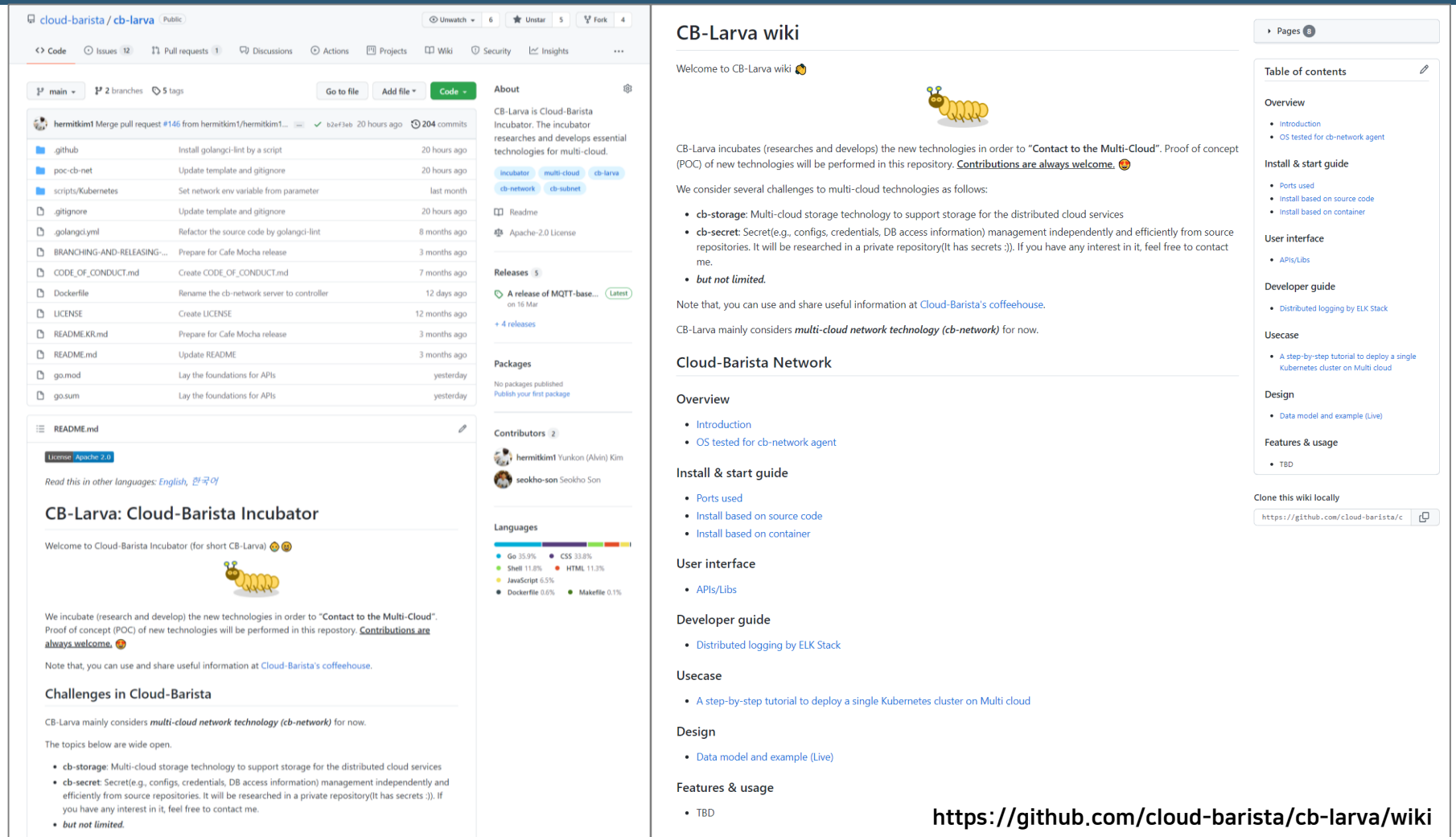


$h(x) = 0.4509x^2 + 0.4243x$

(참고) 공개SW 함께 만들어 갑니다 ^^

cloud-barista/cb-larva: Cloud Adaptive Network 개발 및 정보 공개를 위한 저장소

CB-Larva



https://github.com/cloud-barista/cb-larva/wiki

(참고) 입문자를 위한 플레이그라운드 - Cloud-Barista Coffeehouse

Everything is a contribution! 코드, 정보, 문서, 리뷰, 의견 모두가 소중한 기여이고 함께 발전합니다

README.md

Supermajor

다른 언어로 읽기: English or 한국어.

- ✓ 입문자를 위한 Playground
- ✓ 공유도 기여도 마음이 편한 것이 우선
- ✓ 정보는 다각도로 다양하게 기여 받음

Cloud-Barista's Coffeehouse

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

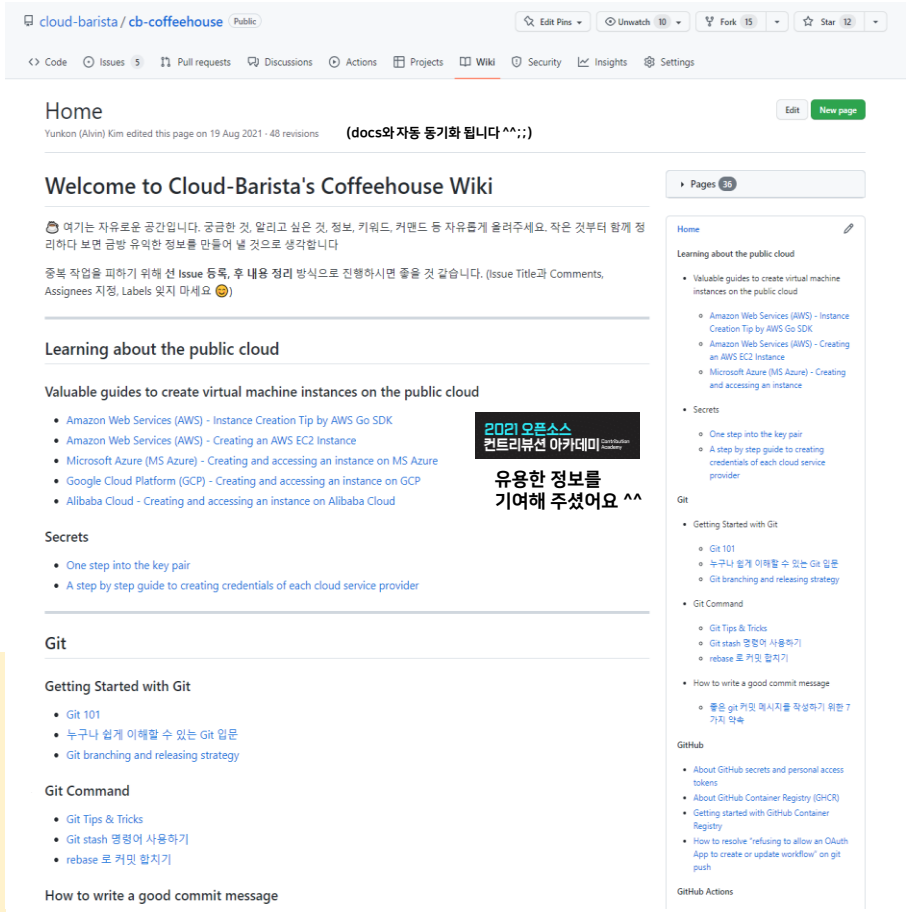
> The historian Brian Cowan describes English coffeehouses as *"places where people gathered to drink coffee, learn the news of the day, and perhaps to meet with other local residents and discuss matters of mutual concern."* (See English coffeehouses in the 17th and 18th centuries)

English Coffeehouse (별칭 Penny House)는 1 Penny가격의 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 찾으시며, Contributor, Reviewer, Maintainer로 거듭나십시오! 😆

Link: https://github.com/cloud-barista/cb-coffeehouse



각종 정보 28건: Cloud-Barista, public cloud, Kubernetes, GitHub, GitHub Actions, Golang, gRPC 등
자동화 스크립트 5건: Cloud-barista, Golang, Docker, Elastic Stack 관련

입문자 시각의 쉬운 설명은 미래 Cloud-Barista Contributor에게 큰 도움이 될 것 입니다 ^^

Topics

Multi-cloud network

Multi-cloud storage

Event-driven architecture

Micro-service architecture

Golang

Web frontend

Distributed key-value store

Distributed storage

[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

클라우드바리스타들의 일곱번째 이야기

멀티클라우드, 컴퓨팅 인프라에 제약없는 서비스 생태계

Cloud-Barista Community the 7th Conference

감사합니다.

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

김 윤 곤 / contact-to-cloud-barista@googlegroups.com