CLOUD BARISTA

멀티/분산 클라우드, 차세대 클라우드를 향한 도전과 기회

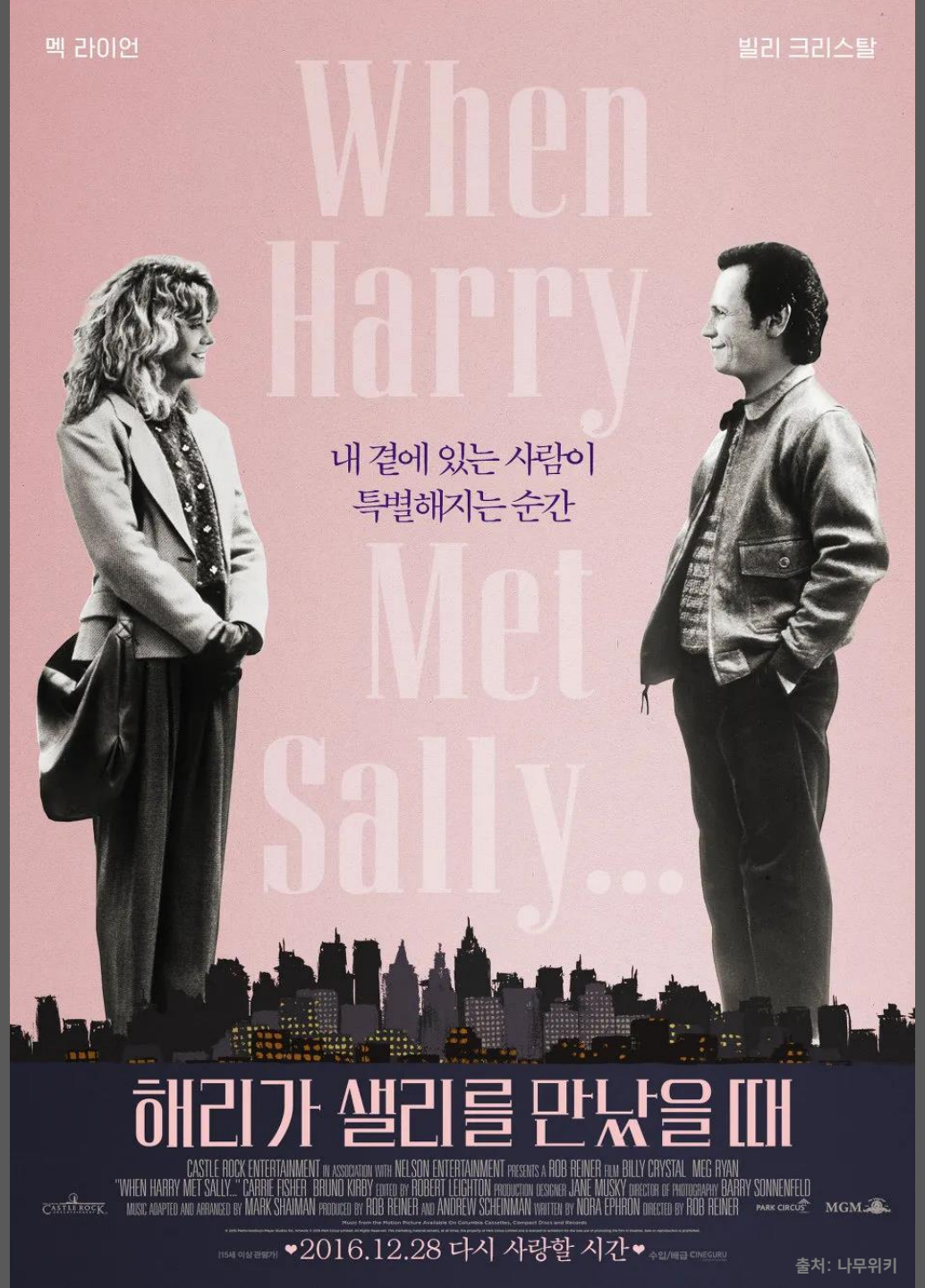
- 클라우드바리스타 커뮤니티 제9차 컨퍼런스 -

Cloud-Barista가 OpenTofu를 만났을 때

메인테이너 @ 클라우드바리스타 커뮤니티

김 윤 곤

시나몬 (Cinnamon) 한잔 어떠세요 ?


멕 라이언
빌리 크리스탈
When Harry Met Sally...
내 곁에 있는 사람이
특별해지는 순간
해리가 샐리를 만났을 때
CASTLE ROCK ENTERTAINMENT IN ASSOCIATION WITH NELSON ENTERTAINMENT PRESENTS A ROB REINER FILM BILLY CRYSTAL MEG RYAN
"WHEN HARRY MET SALLY..." CARRIE FISHER BRUNO KIRBY EDITED BY ROBERT LEIGHTON PRODUCTION DESIGNER JANE MUSKY DIRECTOR OF PHOTOGRAPHY BARRY SONNENFELD
MUSIC ADAPTED AND ARRANGED BY MARK SHAIMAN PRODUCED BY ROB REINER AND ANDREW SCHEINMAN WRITTEN BY NORA EPHRON DIRECTED BY ROB REINER
PARK CIRCUS
MGM
Music from the Motion Picture Available On Columbia Cassettes, Compact Discs and Records
15세 이상 관람가
♥2016.12.28 다시 사랑할 시간♥
수입/배급 CINEGURU

출처: 나무위키

Prologue – Cloud-Barista가 OpenTofu를 만났을 때

졸업 후 멀티 클라우드 기술 여행을 함께 하게 된

Cloud-Barista와 OpenTofu는 성격도 취향도 맞지 않아 헤어지게 되었어요.

(그때는 Terraform 입니다 ^^;;)

이후로, Cloud-Barista는 "멀티 클라우드 네트워크 자원 통합 운용/관리 기술"이 필요 했어요.

- ✓ 필요 사항: 각 클라우드의 기본 네트워크 자원/서비스 연동 및 제어 (CRUD) – VPC, VNet, subnet, public IP, network interface, etc.
- ✗ 필요 사항: 각 클라우드의 연결성 관련 네트워크 자원/서비스 연동 및 제어 (CRUD) – Gateways, routers, routing tables, etc. (예, VPN)
- ✗ 필요 사항: 클라우드간 연결을 위한 정보 및 절차 – Autonomous System Number(ASN), BGP session, connection, secret, etc.

그러나… 2025년에 공통 API가 제공될 예정이었어요. 각 클라우드의 연결성 관련 네트워크 자원/서비스가 추상화된 이후일 것 이었어요…

그러던 Cloud-Barista가 우연히 OpenTofu를 재회하게 되는데…

목 차

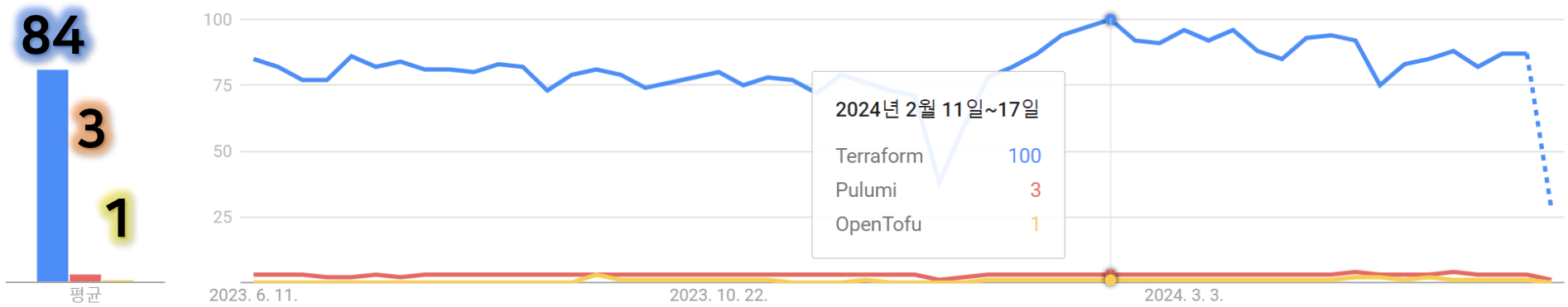
!


HashiCorp
Terraform

?


OpenTofu

시간 흐름에 따른 관심도 변화



#지난12개월 #전세계

#웹검색 #모든카테고리

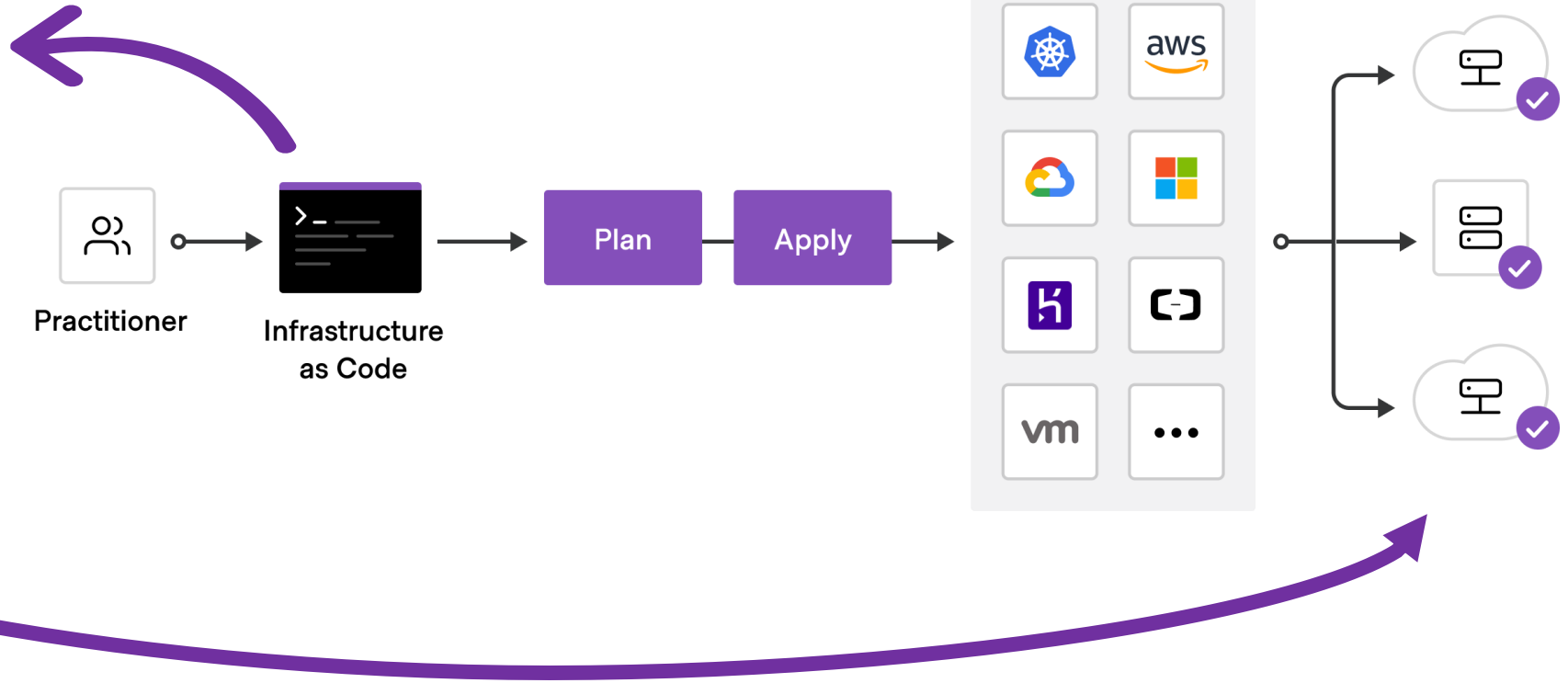
- Terraform 검색어
- Pulumi 검색어
- OpenTofu 검색어

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}

provider "aws" {
  region  = "us-west-2"
}

resource "aws_instance" "app_server" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance"
  }
}
```

HashiCorp Terraform

Terraform is HashiCorp's Infrastructure as Code (IaC) tool.

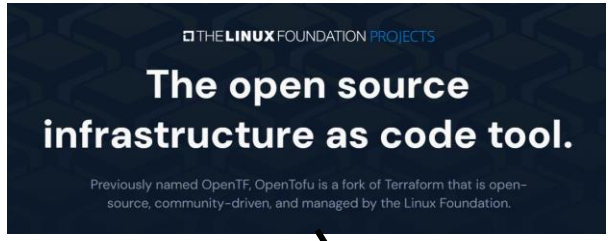
It allows you to manage infrastructure with configuration files rather than through a graphical user interface.



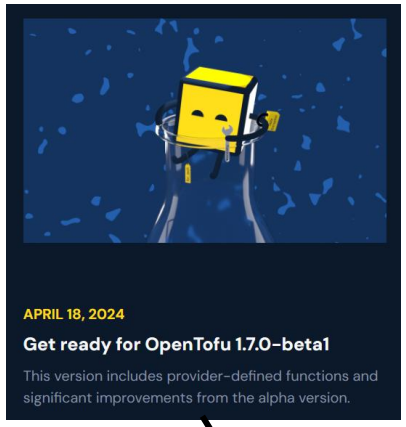
https://developer.hashicorp.com/terraform/tutorials/aws-get-started/infrastructure-as-code

MPL-2.0 license

2023.08.25



2023.09.20



2024.04.18

ORACLE — OPENTOFU — TERRAFORM — HASHICORP — NEWS

Oracle goes vegan: Dumps Terraform for OpenTofu

Oracle E-Business Suite tells customers to shift to latest update built on open source. We guess OpenTofu is enterprise-ready!

THE STACK
May 14, 2024 . 6:00 PM — 2 min read

2024.05.14

2023.08.10

COMPANY

HashiCorp adopts Business Source License

HashiCorp adopts the Business Source License to ensure continued investment in its community and to continue providing open, freely available products.

AUG 10 2023 | ARMON DADGAR

1. What did HashiCorp announce today (Aug 10)?

HashiCorp announced a transition from the Mozilla Public License v2.0 (MPL 2.0) to the Business Source License (BSL, or BUSL) v1.1 for future releases of all products and several libraries. HashiCorp APIs, SDKs, Terraform providers, and almost all other libraries will remain MPL 2.0.

LAST UPDATED: AUGUST 21, 2023

2023.10.27

2024.04.24

IBM

IBM Newsroom | News | Media resources | Inside IBM | Blog

IBM to Acquire HashiCorp, Inc. Creating a Comprehensive End-to-End Hybrid Cloud Platform

$6.4 billion acquisition adds suite of leading hybrid and multi-cloud lifecycle management products to help clients grappling with today's AI-driven application growth and complexity

HashiCorp's capabilities to drive significant synergies across multiple strategic growth areas for IBM, including Red Hat, watsonx, data security, IT automation and Consulting

As a part of IBM, HashiCorp is expected to accelerate innovation and enhance its go-to-market, growth and monetization initiatives

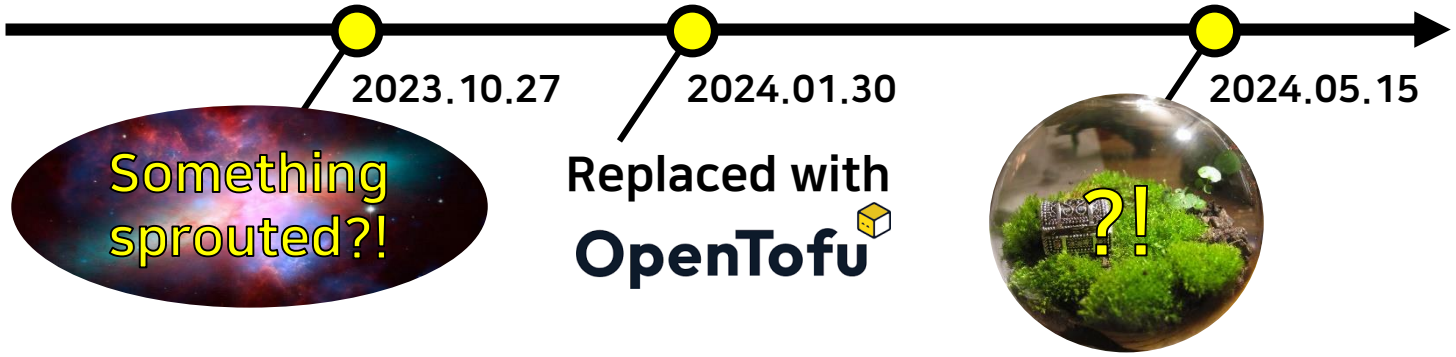
Transaction expected to be accretive to Adjusted EBITDA within the first full year, post close, and free cash flow in year two

Apr 24, 2024

https://www.hashicorp.com/blog/hashicorp-adopts-business-source-license
https://www.hashicorp.com/license-faq
https://opentofu.org/blog/opentofu-announces-fork-of-terraform/
https://github.com/opentofu/opentofu/issues/296
https://www.linuxfoundation.org/press/announcing-opentofu
https://newsroom.ibm.com/2024-04-24-IBM-to-Acquire-HashiCorp-Inc-Creating-a-Comprehensive-End-to-End-Hybrid-Cloud-Platform
https://www.thestack.technology/oracle-dumps-terraform-for-opentofu/

A project for proof of concept (POC) has begun.

poc-mc-net-tf: multi-cloud network with Terraform



Initial commit

yunkon-kim committed 8 months ago

Oct 27, 2023, 10:40 AM GMT+9

Configure and test HA VPN tunnels with OpenTofu

```
* Define providers, which are AWS and GCP
* Create network resources for HA VPN tunnels
* Create EC2/VM instances for ping test
* Create security group or firewall for instance access and testing
```

yunkon-kim committed 5 months ago

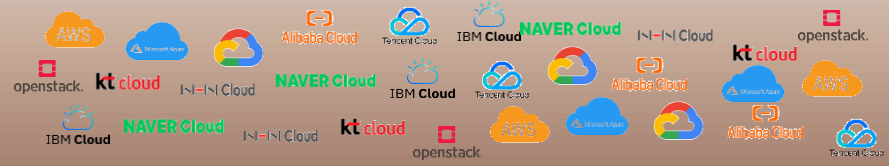
Jan 30, 2024, 7:55 PM GMT+9

When Cloud-Barista Met OpenTofu··· ^^;;

Can
two friends
work
together
and
still maintain
the infrastructure's
harmony
in the
morning?

CLOUD BARISTA

OpenTofu



https://github.com/cloud-barista/mc-terrarium
https://github.com/cloud-barista/mc-terrarium/discussions/59

Cloud-Barista와 OpenTofu가

서로 간에 알아가야 했던 것

1. Cloud-Barista에서 생성한 Multi-cloud infra에, OpenTofu를 활용해 VPN 연결을 보강할 수 있는가 ? Yes
   - 테스트 완료) GCP-AWS간 VPN 연결 보강, GCP-Azure간 VPN 연결 보강

2. Cloud-Barista와 OpenTofu 각각에서 생성한 자원은 각자가 제어 및 관리 할 수 있는가 ? Partially yes
   - 예) Cloud-Barista에서 생성, 운용 및 관리되는 자원이 OpenTofu에서 삭제되면 안됨

3. REST API를 통해 OpenTofu를 제어 및 운용할 수 있는가? Yes
   - 예) Cloud-Barista의 서브시스템/프레임워크과 OpenTofu를 연동하기 위한 REST API 개발 및 제공

4. OpenTofu의 Human-readable 입/출력 대신에 Machine-readable 입/출력을 할 수 있는가 ? Yes (on-going)

5. Cloud-Barista와 OpenTofu을 활용하여 상호 보완적인 자원 운용/관리가 가능한가 ? Conditionally yes

(시작은 Terraform이었으나… ^^;;)

OpenTofu와 첫 만남은… 강렬했다…

VPN으로 시작해서…

온라인 상에 다양한 정보

6개월 ~ 5년 이전 정보

여러 차례 버전이 개선되었고

CSP 인증 메커니즘이 바뀌었고

대부분 그대로 동작이 되지 않음…

하얗게 불타버렸다…

IaC를 처음 접하는 초보자

한 CSP의 문법도 제대로 모름

양 CSP간 연결성 디버깅 불가

믿었던 ChatGPT 너마저…

OTL…
Terraform에 대해서는…
뱁새였어…

기본적인 것부터! 튜토리얼 및 Docs 뜯어보기

AWS Tutorial 및 Docs를 살펴보는 시간 ➔ OpenTofu의 인프라 구분/관리 기준 파악할 수 있었던 의미 있는 시간
(모듈/리소스를 구분 또는 관리하는 기준)

```
terraform {                          Terraform block
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}

provider "aws" {                     Provider block
  region  = "us-west-2"
}

                                     Resource block
resource "aws_instance" "app_server" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance"
  }
}
```

TF configuration 예

AWS Tutorial

➔ 기본 지식 습득 및 검증

- Infrastructure 형상을 작성 (IaC)
- 여러가지 Block들의 특성 파악
- Resource 간에 Value 참조 방식 파악

➔ Tofu CLI 동작 메커니즘 및 입/출력 파악

- 인프라 초기화 tofu init
- 인프라 검증 tofu plan
- 인프라 생성 tofu apply
- 인프라 삭제 tofu destroy
- 기타 등등

Directories and Modules

File 기반

Directory 기반

A *module* is a collection of `.tf` and/or `.tf.json` files kept together in a directory.

A Terraform module only consists of the top-level configuration files in a directory; nested directories are treated as completely separate modules, and are not automatically included in the configuration.

https://developer.hashicorp.com/terraform/language/files#directories-and-modules

1. 자체 작업 Directory 필요
   - 작업 디렉토리에 Terraform configuration (TF configuration) files 위치
2. TF configuration files 작성 필요
   - TF configurations는 인프라를 정의/명세하는 Files 세트를 나타냄
   - 파일 세트 예: *.tf, *.tf.json, *.tf.var, *.tf.var.json

인프라 구분/관리 체계 마련 ※ 본 프로젝트의 중심이 되는 구조

디렉토리 기반의 관리 체계 마련: 한 디렉토리에, 인프라 정의/명세 파일들 작성 → OpenTofu CLI 실행 시 명세된 목표 상태에 도달
(TF configuration: *.tf, *.tf.json, *.tf.var, *.tf.json) (선언형)

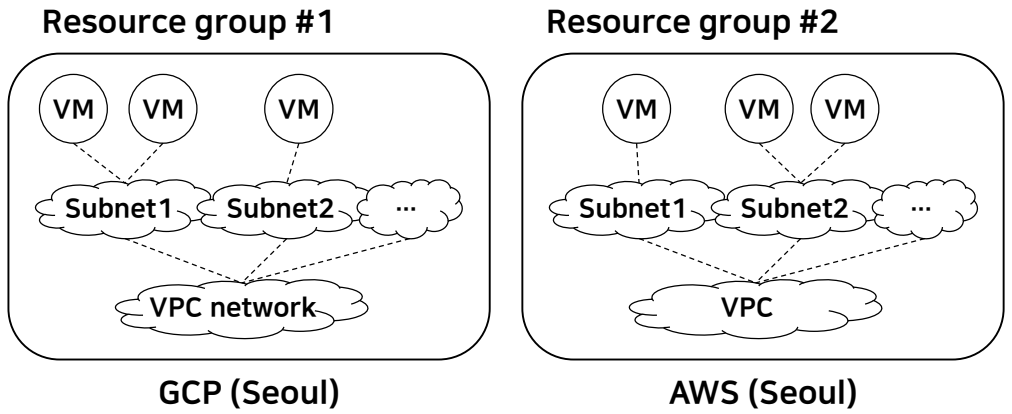
자유롭게 Resource Group ID를 부여하도록 설계 ← 인프라 관리자의 의도대로 리소스들을 그룹핑할 수 있기 때문

- .terrarium directory ← 인프라 (TF configuration files) 운용 및 관리
  - {resourceGroupId} directory
    - vpn
      - gcp-aws
      - gcp-azure
      - ...

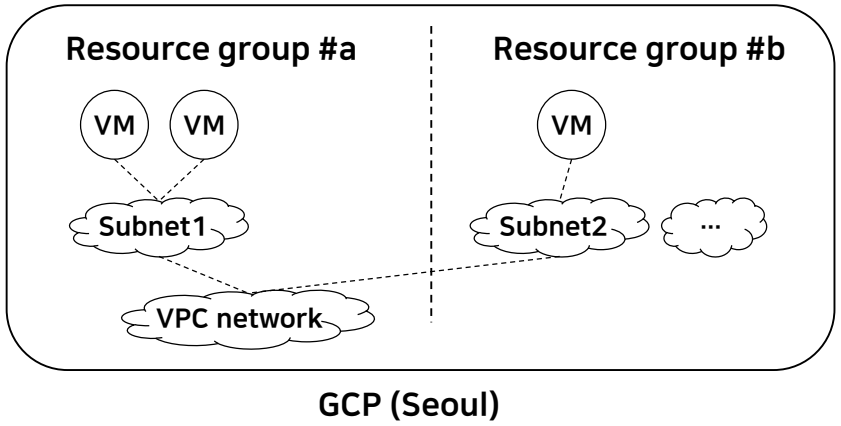
    → 인프라 운용 및 관리 "환경" + "보강"할 인프라 자원
  - {resourceGroupId} directory
    - vpn
      - gcp-aws
      - gcp-azure
      - ...

    → 인프라 운용 및 관리 "환경" + "보강"할 인프라 자원
- secrets directory ← CSP 관련 secrets 관리
  - 각 CSP의 credential 파일을 관리
- templates directory ← TF configuration templates 관리
  - vpn
    - gcp-aws
    - gcp-azure
    - ...

예시1



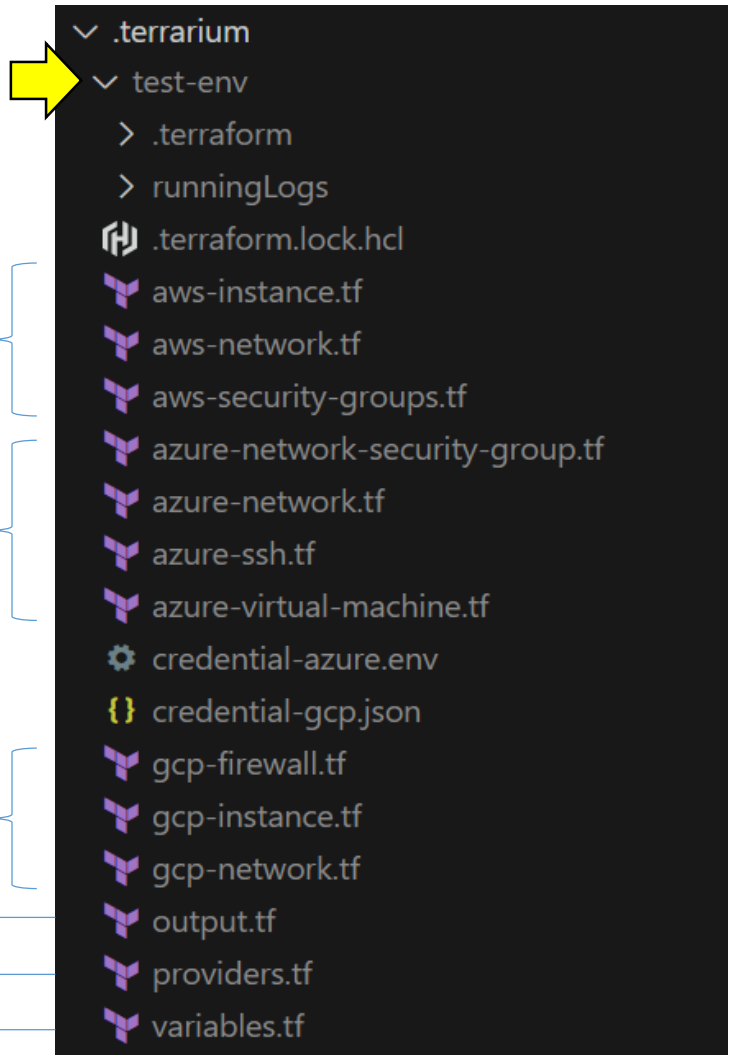
예시2



예시) 인프라 운용 및 관리 “환경” 및 “보강”할 인프라 자원

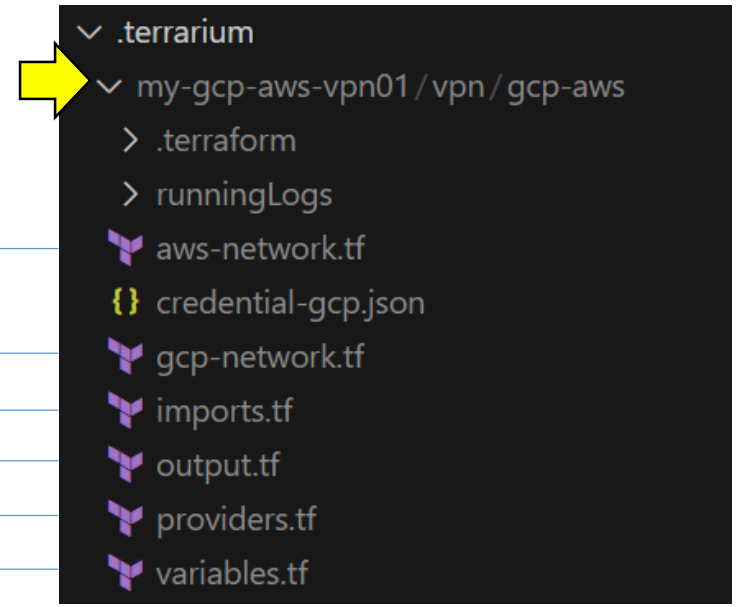
인프라 운용 및 관리
“환경1”
ID: test-env

“보강”할 인프라 자원
연구 개발 인프라(테스트 용)



인프라 운용 및 관리
“환경2”
ID: my-gcp-aws-vpn-01

“보강”할 인프라 자원
GCP-AWS VPN 보강



(참고) Tumblebug의 자원 관리 체계와 호환 가능

✓ 자체적/독립적으로도 활용 가능

- .terrarium directory ← 자원 (TF configurations) 운용 및 관리
  - {resourceGroupId} directory
    - vpn
      - gcp-aws
      - gcp-azure
      - ...
  - {resourceGroupId} directory
    - vpn
      - gcp-aws
      - gcp-azure
      - ...
- secrets ← CSP 관련 secrets 관리
  - 각 CSP의 credential 파일을 관리
- templates ← TF configuration template 관리
  - vpn
    - gcp-aws
    - gcp-azure
    - ...

활용 예:

resourceGroupId = nsId + mcisId

- 예) "rgId": "ns01-mcis01"
- 이슈: ns01-mcis01에 1개의 gcp-aws tunnel 자원 생성 가능

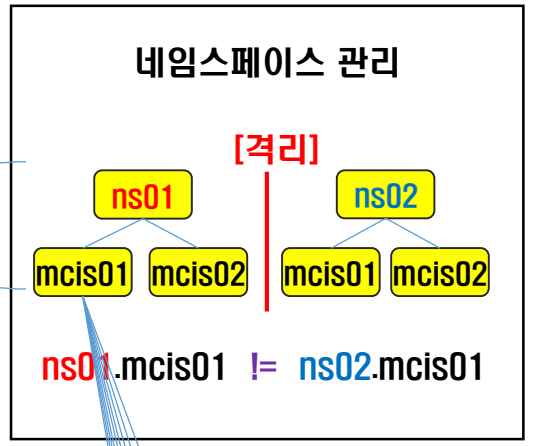
resourceGroupId = nsId + mcisId + GUID

- 예) "rgId": "ns01-mcis01-GUID"
- 이슈: 사용자가 rgId를 관리해야함

이슈:

- CSP와 CB에서 관리하는 ID 체계가 상이하나,
  Tofu는 CSP에서 관리하는 ID/NameID 활용하고 있음

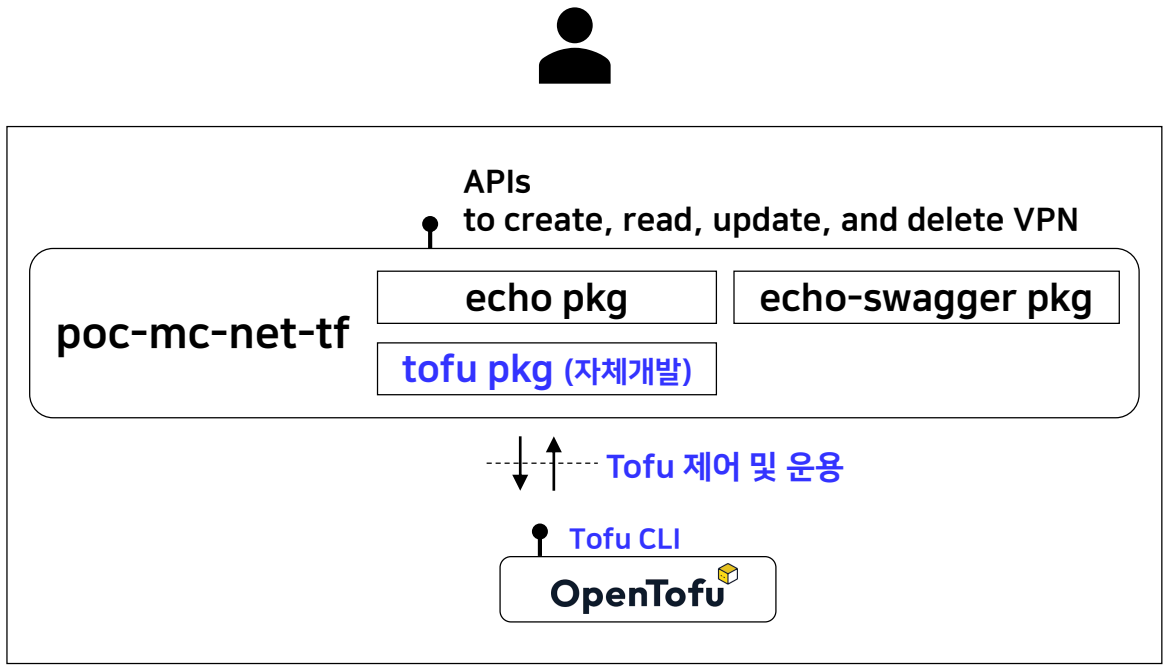
Tumblebug 자원 관리 체계



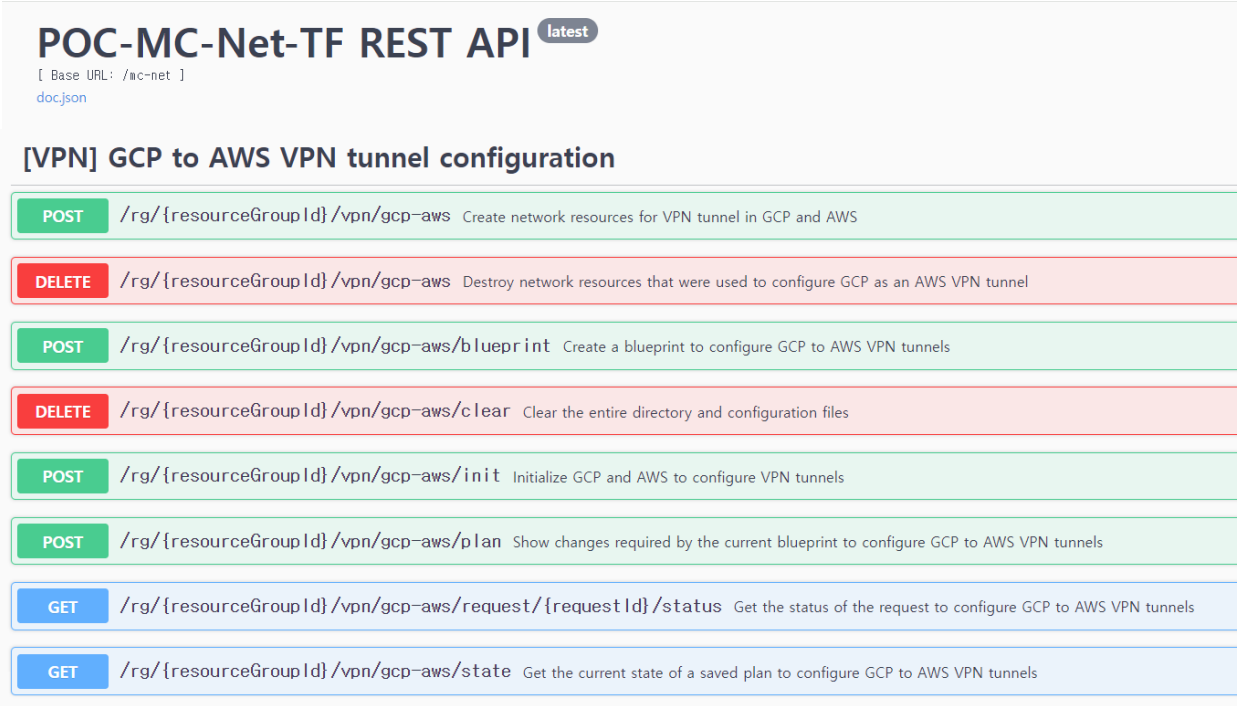
- Network
- Spec
- Image
- Access key
- Security group
- Snapshot and custom image
- NLB
- Data disk
- Cluster

OpenTofu 제어 및 운용을 위한 REST API 제공

사용자가 API 요청 시, Tofu CLI 명령을 연계/수행할 수 있도록 tofu pkg 자체 개발



시스템 구조 in a host/container



REST API sample

API를 활용한 인프라 생성 시, Customization 필수!

Challenge: 정적인 TF configuration 파일에 인프라가 정의/명세되어 있는데, 어떻게 Customization하죠?

```
terraform {                          Terraform block
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}

pr

}

resource "aws_instance" "app_server" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance"
  }
}
```

예를 들어, 사용자가
이미지, 스펙을 변경하려면?

Resource block

TF configuration 예

Variable block을 활용한 TF configuration의 Template화 추진 및 제공

Example: variable block for input

```
variable "user_information" {        Variable block
  type = object({
    name    = string
    address = string
  })
  sensitive = true
}

resource "some_resource" "a" {
  name    = var.user_information.name
  address = var.user_information.address
}
```

Variable Definitions (`.tfvars`) Files

To set lots of variables, it is more convenient to specify their values in a *variable definitions file* (with a filename ending in either `.tfvars` or `.tfvars.json`) and then specify that file on the command line with `-var-file`:

Variable Definition Precedence

Terraform loads variables in the following order, with later sources taking precedence over earlier ones:

- Environment variables
- The `terraform.tfvars` file, if present.
- The `terraform.tfvars.json` file, if present.
- Any `*.auto.tfvars` or `*.auto.tfvars.json` files, processed in lexical order of their filenames.
- Any `-var` and `-var-file` options on the command line, in the order they are provided. (This includes variables set by an HCP Terraform workspace.)

Example: input validation

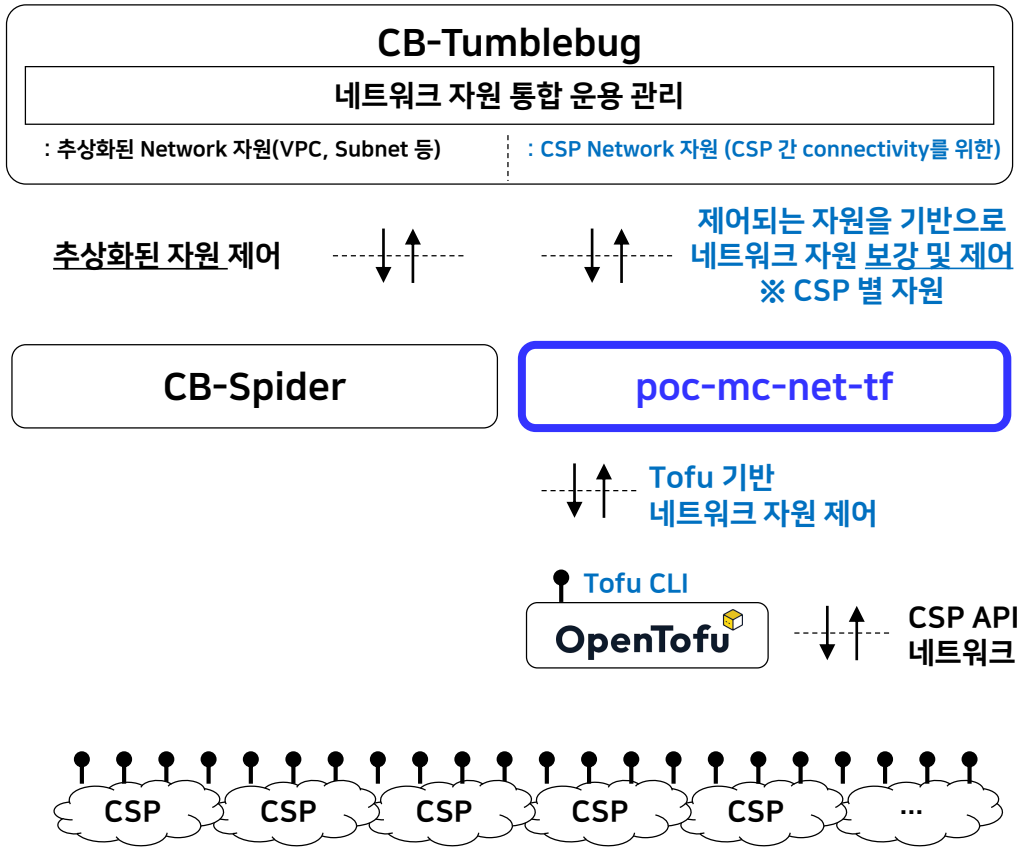
```
validation {                         Validation
  condition     = length(var.image_id) > 4 && substr(var.image_id, 0, 4) == "ami-"
  error_message = "The image_id value must be a valid AMI id, starting with \"ami-\"."
}
```

https://developer.hashicorp.com/terraform/language/values/variables

poc-mc-net-tf 포지셔닝, Cloud-Barista 서브시스템과 연계성

- Supported Computing Infrastructure Resources
  - **Basic Resources**: Public Image, VM Spec, VPC/Subnet, Security Group, VM KeyPair
  - **VM Infrastructures**: VM, NLB(Network Load Balancer), Disk, MyImage
  - **Container Infrastructures**: PMKS(Provider-Managed K8S)
- Supported CloudOS:

Provider, CloudOS	CloudOS Constant	Cloud Driver Lib.	Etc
Amazon Web Services	AWS	aws-driver-v1.0.so	
Microsoft Azure	AZURE	azure-driver-v1.0.so	
Google Cloud Platform	GCP	gcp-driver-v1.0.so	
Alibaba Cloud	ALIBABA	alibaba-driver-v1.0.so	
Tencent Cloud	TENCENT	tencent-driver-v1.0.so	
IBM VPC Cloud	IBM	ibmvpc-driver-v1.0.so	
OpenStack Platform	OPENSTACK	openstack-driver-v1.0.so	
NCP Classic Cloud	NCP	ncp-driver-v1.0.so	
NCP VPC Cloud	NCPVPC	ncpvpc-driver-v1.0.so	
NHN Cloud	NHNCLOUD	nhncloud-driver-v1.0.so	
KT Classic Cloud	KTCLOUD	ktcloud-driver-v1.0.so	
KT VPC Cloud	KTCLOUDVPC	ktcloudvpc-driver-v1.0.so	



※ 파란색 부분: POC 영역

[부연설명]
* Tumblebug이 Spider를 통해 VPC 생성
  - 생성된 VPC는 기존과 같이 Tumblebug이 관리 및 제어
Tumblebug이 poc-mc-net-tf를 통해 VPN Tunnel 구성
  - 생성된 VPN tunnel 자원은 poc-mc-net-tf가 제어 (및 관리)

Resources/services *required* for multi-cloud network configuration

- HA VPN tunnel related resources/services,
- Inter-cloud connectivity related resources/services,
- and so on.

Source:
- CB-Spider, "Supported Computing Infrastructure Resources", (accessed on 2024-03-14, https://github.com/cloud-barista/cb-spider/wiki/Supported-Compute-Infrastructure-Resources#supported-computing-infrastructure-resources)
- CB-Spider, "Supported CloudOS", (accessed on 2024-03-14, https://github.com/cloud-barista/cb-spider/wiki/Supported-CloudOS#supported-cloudos)

개념도: 서브시스템간 연계 및 활용

활용 예

(Tumblebug API / Map UI)
1. MCIS 생성
(VPC, Subnet 등 생성)

(tofu init)
2. Providers 설정

(POST /infracode)
3. VPN tunnel 구성을 위한 TF configuration 생성
* 정보 주입: 앞서 생성한 MCIS에서 연동하려는 두 Site의 정보
* 정보: CSP, Region, vNet, (Subnet), (ResourceGroup), (CIDR block for GW subnet), 등

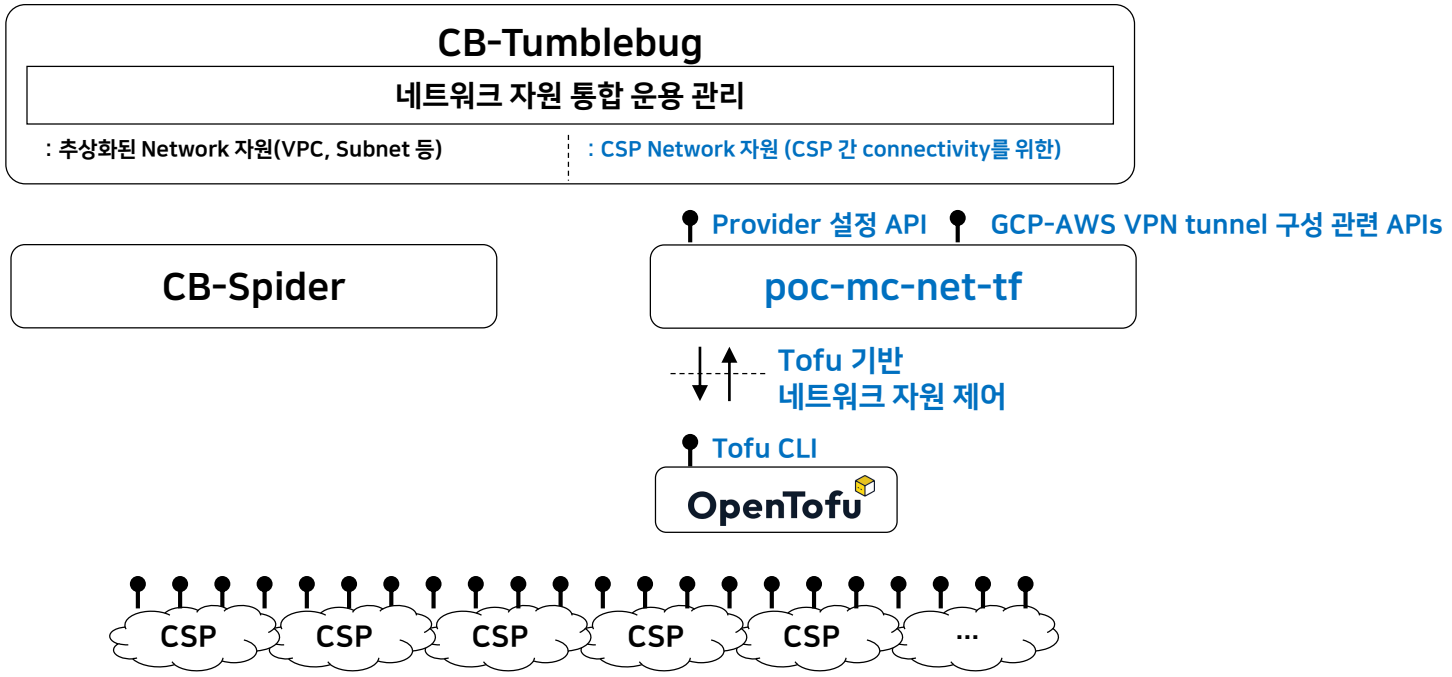
(tofu plan)
4. 구성할 VPN tunnel 검토

(tofu apply)
5. CSP에 VPN tunnel 생성

(tofu state or GET req/{reqId}/status)
6. 조회

(tofu destroy or DELETE /clear)
7, 삭제

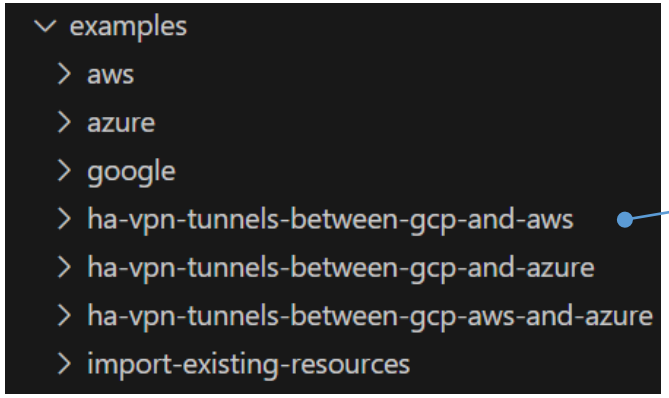
구성도



(참고) 개발/기여 프로세스 정립 (derived from 다사다난했던 개발 과정)

1. 정적 infra 구성 및 테스트

* 온라인 예제/샘플 활용, 테스트, 트러블슈팅
* CSP 콘솔/포털 상에서 인프라 정보/상태 확인 및 검증
* Maintaining 대상 아님

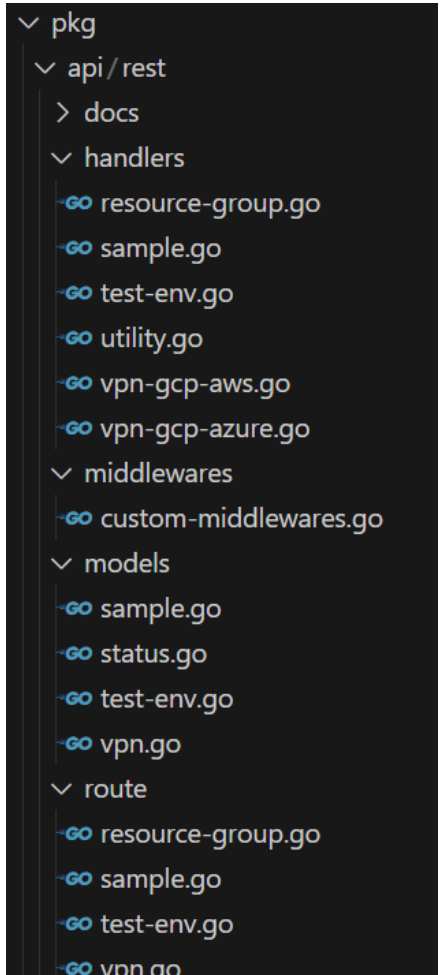


2. Template 개발

* 사용자 입/출력을 Variable/output으로 추출
* Providers/resources 에 적용 및 개선
* Template 기반 인프라 생성 테스트 및 디버깅

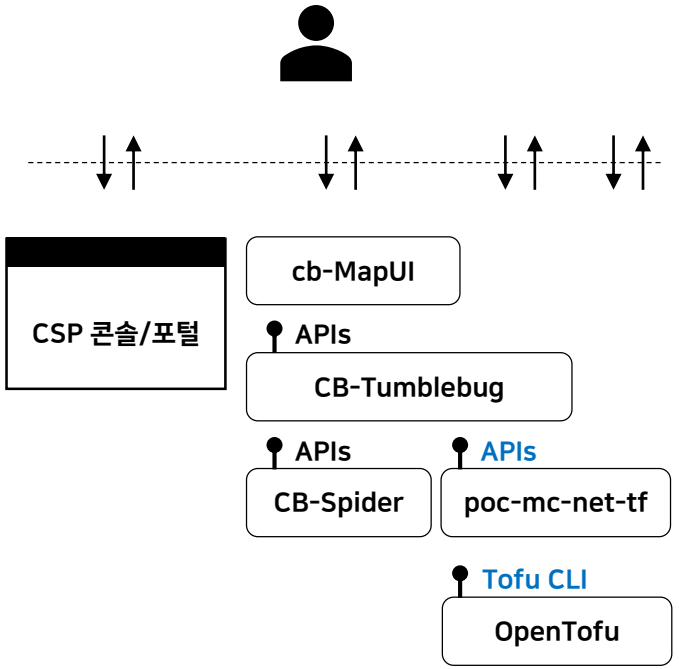
3. APIs 개발

* REST API 개발, 테스트 및 디버깅



4. 연계 테스트

* 서브시스템간 연계 테스트 및 디버깅



A project for proof of concept (PoC) has begun.

Multi-cloud network with Terraform (poc-mc-net-tf)

Renewal!
Introducing mc-terrarium

2023.10.27

2024.01.30

Replaced with OpenTofu

2024.05.15

Initial commit

yunkon-kim committed 8 months ago

Verified 7095b17

Oct 27, 2023, 10:40 AM GMT+9

Configure and test HA VPN tunnels with OpenTofu

```
* Define providers, which are AWS and GCP
* Create network resources for HA VPN tunnels
* Create EC2/VM instances for ping test
* Create security group or firewall for instance access and testing
```

69e0403

yunkon-kim committed 5 months ago

Jan 30, 2024, 7:55 PM GMT+9

Renewal! Introducing mc-terrarium #59

yunkon-kim announced in Announcements

yunkon-kim last month Maintainer edited

영어로 된 원본 주석 - 한국어로 번역

I am pleased to introduce **mc-terrarium: multi-cloud terrarium**.

Having successfully completed the initial development and testing with poc-mc-net-tf, I have determined that it is beneficial to expand and shift the scope and direction of this project.

The mc-terrarium aims to provide an environment to enrich multi-cloud infrastructure. This is gradually evolving to enable you to build the multi-cloud infrastructure you need.

Note - Renewal Schedule: The first phase will be completed by the end of June.

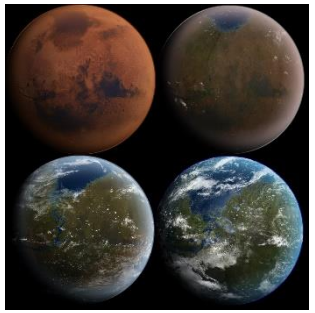
For more information about what a **terrarium** is and some interesting content related to **terra-**, please explore the following links:

▶ Click to meet the articles

↑ 1

https://github.com/cloud-barista/mc-terrarium
https://github.com/cloud-barista/mc-terrarium/discussions/59

Terraforming



- Terraforming or terraformation ("Earth-shaping") is the hypothetical process of deliberately modifying the atmosphere, temperature, surface topography or ecology of a planet, moon, or other body to be similar to the environment of Earth to make it habitable for humans to live on.

HashiCorp Terraform

- Terraform is an infrastructure-as-code software tool created by HashiCorp. Users define and provide data center infrastructure using a declarative configuration language known as HashiCorp Configuration Language (HCL), or optionally JSON.

Terraformer

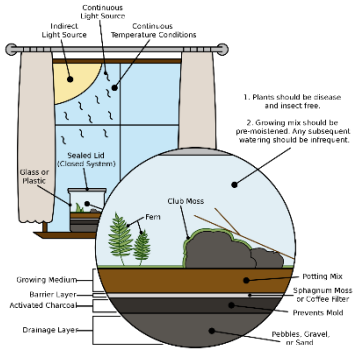
GoogleCloudPlatform / terraformer

terraformer Public

waze SRE

- Terraformer is a CLI tool that generates tf/json and tfstate files based on existing infrastructure (reverse Terraform).
  - Organization: GoogleCloudPlatform
  - Repository: terraformer
  - Disclaimer: This is not an official Google product
  - Created by: Waze SRE

Terrarium



- A terrarium (pl.: terraria or terrariums) is usually a sealable glass container containing soil and plants that can be opened for maintenance to access the plants inside; however, terraria can also be open to the atmosphere.
  Terraria are often kept as ornamental items.

David Latimer's terrarium hasn't been watered in 47 years. Photo: www.dailymail.co.uk

(This article was written in 2019)

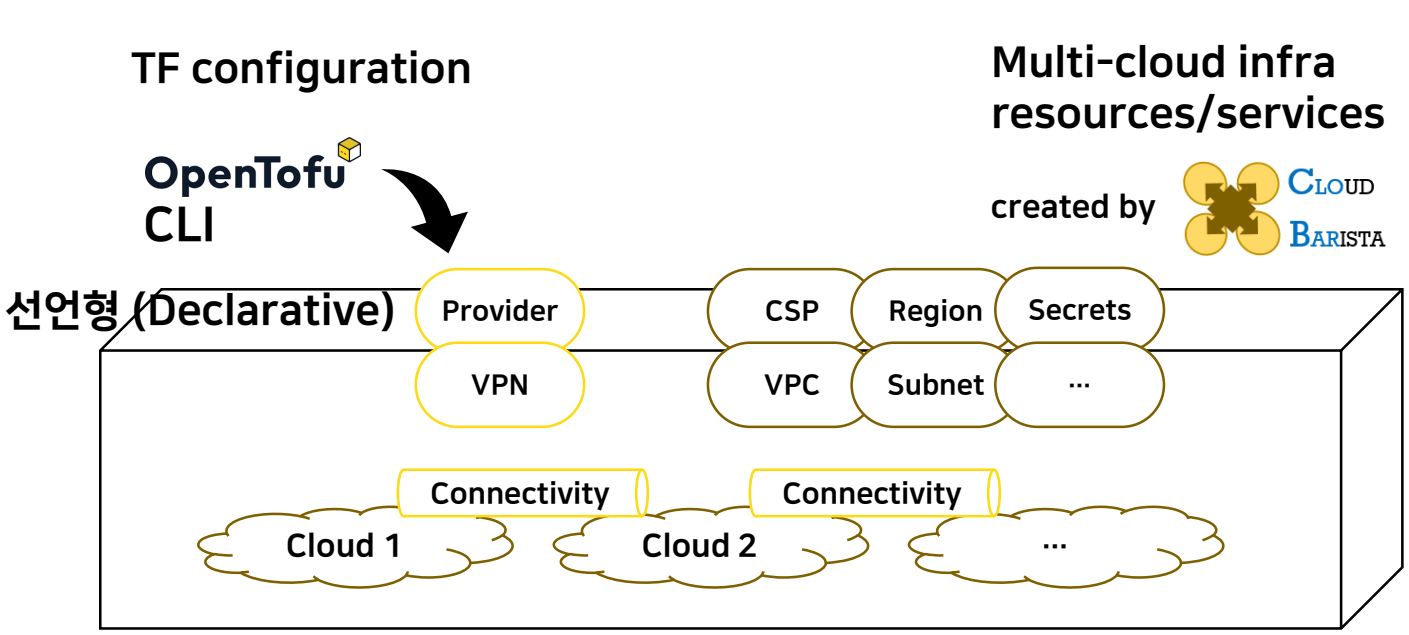
The oldest one is claimed to be one grown by David Latimer of England, started in 1960, when he planted a single tradescantia cutting inside, and last opened in 1972, when he added a bit of water, then sealed again, never to be reopened. That means it has been growing inside its bottle with no additional air, water or fertilizer for 47 years. The same photo has been circulating on the Internet for at least a decade. David has certainly aged since then, but the appearance of the terrarium probably hasn't changed.

Source: https://laidbackgardener.blog/2019/10/14/the-worlds-second-oldest-sealed-terrarium/

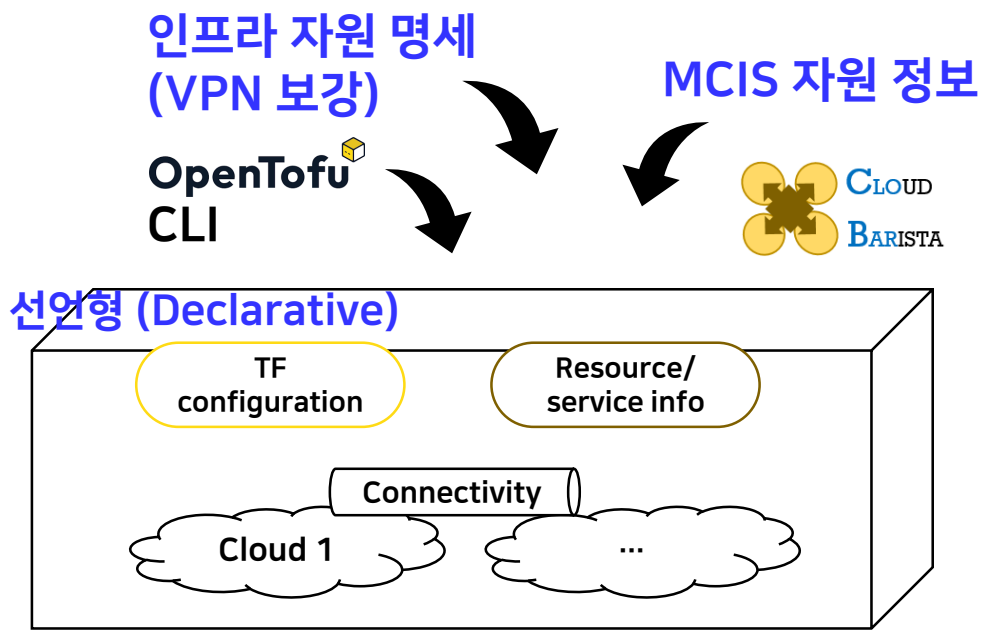
(formerly poc-mc-net-tf)

mc-terrarium: Multi-cloud terrarium

- "mc-terrarium" is an OpenTofu-based tool that
  provides an environment (i.e., terrarium) and features to enrich multi-cloud infrastructure.
- The terrarium consists of
  - resources/services information created by Cloud Barista,
  - TF configuration indicating resource/service information to be added,
  - TofuCLI, etc.

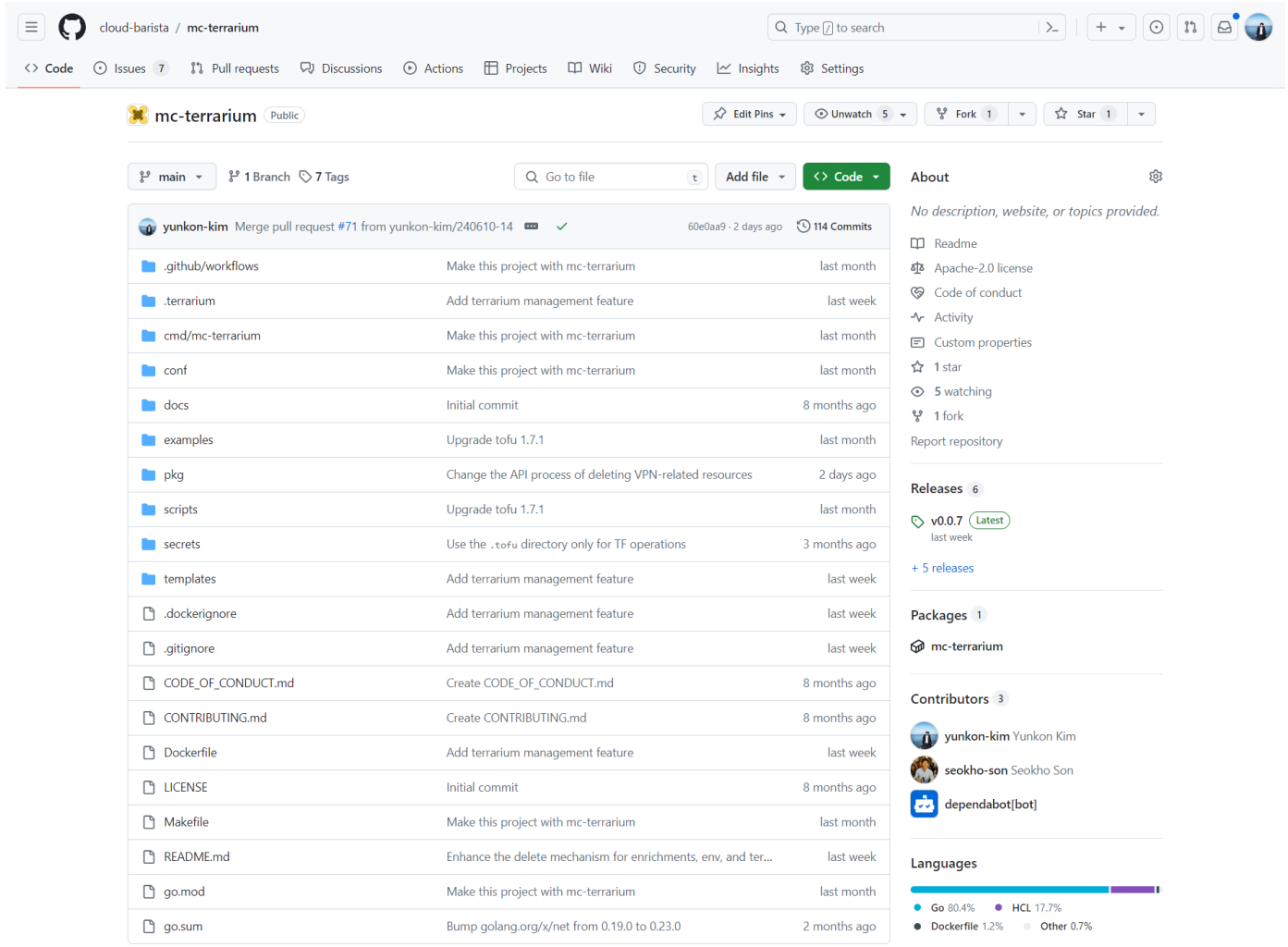


Terrarium for multi-cloud infrastructure



mc-terrarium

mc-terrarium 저장소 및 APIs 소개



https://github.com/cloud-barista/mc-terrarium

[Terrarium] An environment to enrich the multi-cloud infrastructure

GET	/tr	Read all terrarium
POST	/tr	Issue/create a terrarium
GET	/tr/{trId}	Read a terrarium
DELETE	/tr/{trId}	Erase the entire terrarium including directories and configuration files

[VPN] GCP to AWS VPN tunnel configuration

GET	/tr/{trId}/vpn/gcp-aws	Get resource info to configure GCP to AWS VPN tunnels
POST	/tr/{trId}/vpn/gcp-aws	Create network resources for VPN tunnel in GCP and AWS
DELETE	/tr/{trId}/vpn/gcp-aws	Destroy network resources that were used to configure GCP as an AWS VPN tunnel
POST	/tr/{trId}/vpn/gcp-aws/env	Initialize a multi-cloud terrarium for GCP to AWS VPN tunnel
DELETE	/tr/{trId}/vpn/gcp-aws/env	Clear the entire directory and configuration files
POST	/tr/{trId}/vpn/gcp-aws/infracode	Create the infracode to configure GCP to AWS VPN tunnels
POST	/tr/{trId}/vpn/gcp-aws/plan	Check and show changes by the current infracode to configure GCP to AWS VPN tunnels
GET	/tr/{trId}/vpn/gcp-aws/request/{requestId}	Check the status of a specific request by its ID

[VPN] GCP to Azure VPN tunnel configuration (under development)

GET	/tr/{trId}/vpn/gcp-azure	Get resource info to configure GCP to Azure VPN tunnels
POST	/tr/{trId}/vpn/gcp-azure	Create network resources for VPN tunnel in GCP and Azure
DELETE	/tr/{trId}/vpn/gcp-azure	Destroy network resources that were used to configure GCP as an Azure VPN tunnel
POST	/tr/{trId}/vpn/gcp-azure/env	Initialize a multi-cloud terrarium for GCP to Azure VPN tunnel
DELETE	/tr/{trId}/vpn/gcp-azure/env	Clear the entire directory and configuration files
POST	/tr/{trId}/vpn/gcp-azure/infracode	Create the infracode to configure GCP to Azure VPN tunnels
POST	/tr/{trId}/vpn/gcp-azure/plan	Check and show changes by the current infracode to configure GCP to Azure VPN tunnels
GET	/tr/{trId}/vpn/gcp-azure/request/{requestId}	Check the status of a specific request by its ID

Terrarium APIs 요약 설명

[Terrarium] An environment to enrich the multi-cloud infrastructure

GET	/tr	Read all terrarium
POST	/tr	Issue/create a terrarium
GET	/tr/{trId}	Read a terrarium
DELETE	/tr/{trId}	Erase the entire terrarium including directories and configuration files

[VPN] GCP to AWS VPN tunnel configuration

GET	/tr/{trId}/vpn/gcp-aws	Get resource info to configure GCP to AWS VPN tunnels
POST	/tr/{trId}/vpn/gcp-aws	Create network resources for VPN tunnel in GCP and AWS
DELETE	/tr/{trId}/vpn/gcp-aws	Destroy network resources that were used to configure GCP as an AWS VPN tunnel
POST	/tr/{trId}/vpn/gcp-aws/env	Initialize a multi-cloud terrarium for GCP to AWS VPN tunnel
DELETE	/tr/{trId}/vpn/gcp-aws/env	Clear the entire directory and configuration files
POST	/tr/{trId}/vpn/gcp-aws/infracode	Create the infracode to configure GCP to AWS VPN tunnels
POST	/tr/{trId}/vpn/gcp-aws/plan	Check and show changes by the current infracode to configure GCP to AWS VPN tunnels
GET	/tr/{trId}/vpn/gcp-aws/request/{requestId}	Check the status of a specific request by its ID

mc-terrarium 관리

1. Terrarium ID (trId) 생성

GCP to AWS VPN 보강

5. 인프라 보강 수행 (자원 생성)

2. 환경 구성/초기화

3. Infracode (TF configuration) 생성

4. Infracode 검증

trId * required
string
(path)

Terrarium ID
Default value : tr01
tr01

ParamsForInfracode * required
object
(body)

Parameters requied to create the infracode to

Example Value | Model

```
{
  "tfVars": {
    "aws-region": "ap-northeast-2",
    "aws-subnet-id": "subnet-xxxxx",
    "aws-vpc-id": "vpc-xxxxx",
    "gcp-region": "asia-northeast3",
    "gcp-vpc-network-name": "tr-gcp-vpc",
    "terrarium-id": ""
  }
}
```

Parameter content type
application/json

x-request-id
string
(header)

Custom request ID
x-request-id

➔ ReqBody/입력?
보강될 인프라 정보
(e.g., MCIS)

➔ Templates ?
보강할 자원
(e.g., GCP-AWS VPN)

- templates
  - vpn
    - gcp-aws
      - aws-network.tf
      - gcp-network.tf
      - imports.tf
      - output.tf
      - providers.tf
      - variables.tf
    - gcp-azure
      - azure-network.tf
      - gcp-network.tf
      - output.tf
      - providers.tf
      - variables.tf

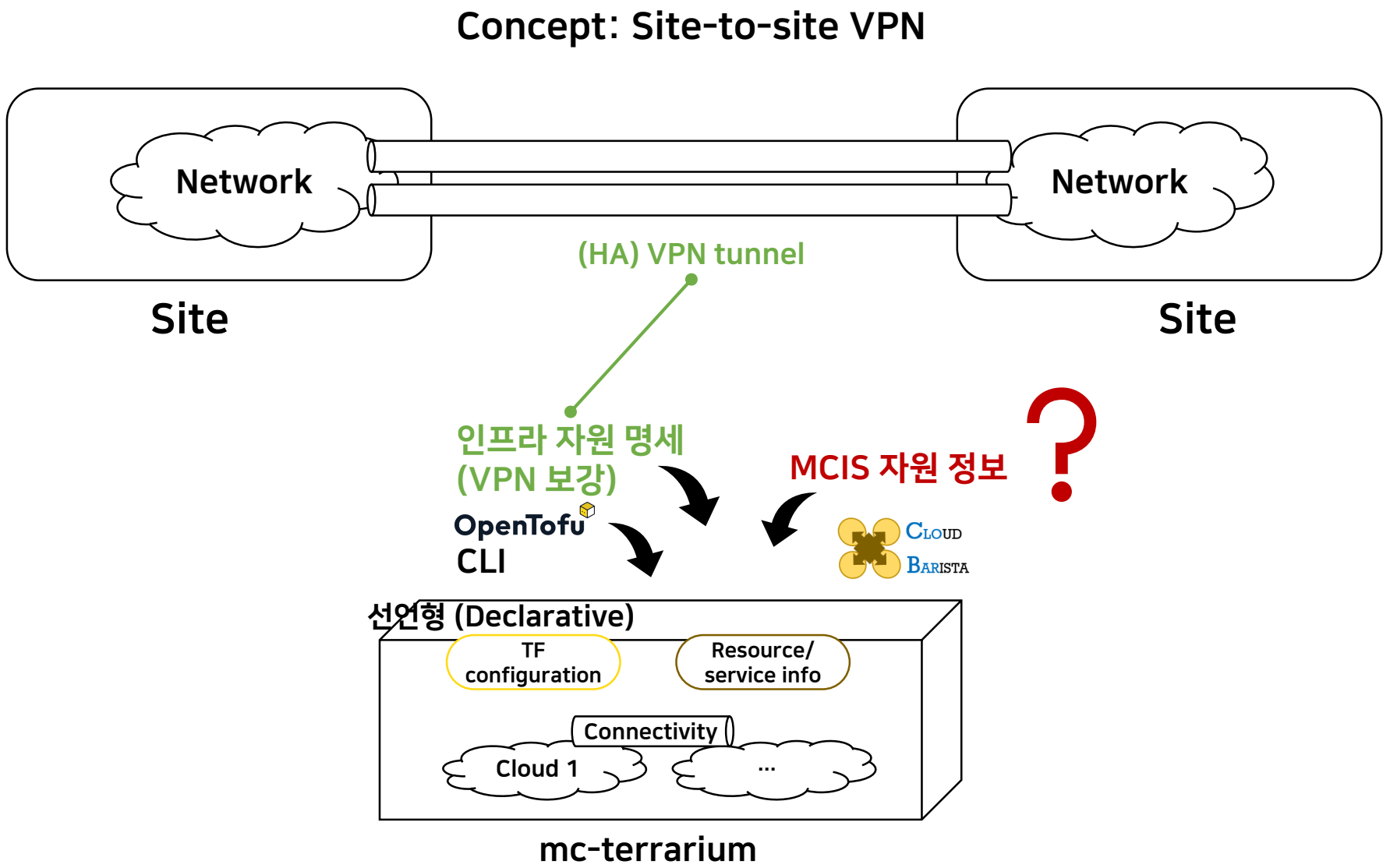
mc-terrarium을 통해 생성되는

VPN 관련 네트워크 자원/서비스 및 연결 정보

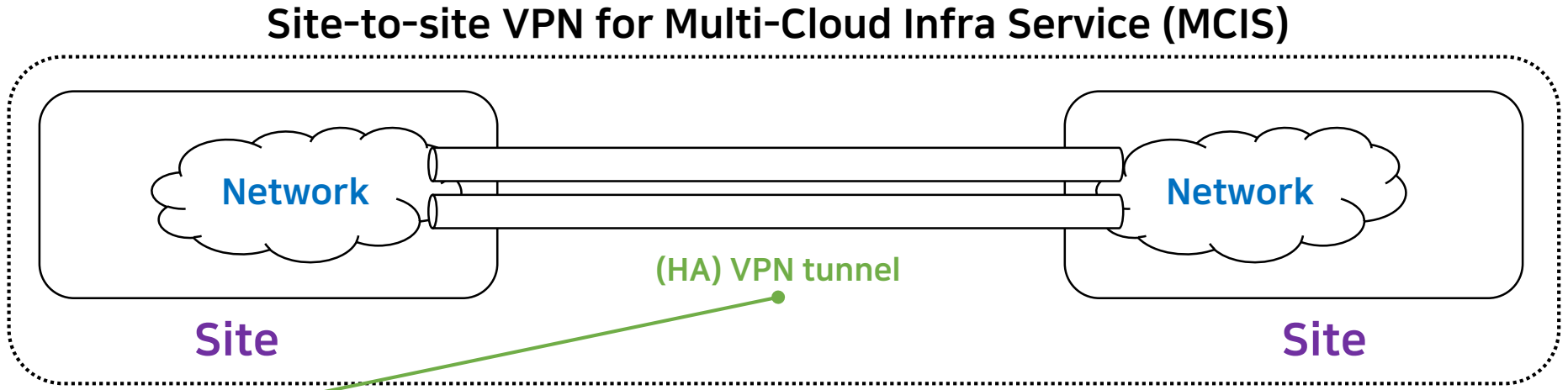
- VPN tunnel을 위해 생성되는 다양한 CSP 네트워크 자원/서비스:
  - GCP (7종 자원):
    - Names of compute router, HA VPN gateway, external VPN gateway (1) and interfaces (2~4), VPN tunnels (2~4), router interfaces (2~4), router peers (2~4)
  - AWS (3종 자원):
    - Tag names of VPN gateway, customer gateways (2), VPN connections (2)
  - Azure (5종 자원 + 부가 정보):
    - Names of public IPs (2), virtual network gateway(1) and IP configurations (2), local gateways (2), network gateway connections (2)
    - Automatic Private IP Addressing (APIPA) addresses (2)
    - A list of allowed Stock Keeping Unit (SKU) (e.g., ["VpnGw1AZ", "VpnGw2AZ", "VpnGw3AZ", "VpnGw4AZ", …])
    - A VPN SKU (e.g., VpnGw1: The Azure VPN Sku/Size)
    - A preshared-secret for the VPN connection (e.g., 1234567890)

- (예정) 사용자가 원하는 Autonomous System Number(ASN)를 입력하도록 개선
  - Autonomous System Number(ASN)은 Border Gateway Protocol(BGP)에 사용되는 번호임
  - ASN은 1에서 65534 사이의 고유한 16비트 숫자 또는 131072와 4294967294 사이의 32비트 숫자

Site-to-site VPN (← GCP to AWS VPN / GCP to Azure VPN)



Site-to-site VPN과 CB-Tumblebug



인프라 자원 명세
(VPN 보강)

MCIS 자원 정보

OpenTofu
CLI

선언형 (Declarative)

TF configuration

Resource/ service info

Connectivity

Cloud 1

...

mc-terrarium

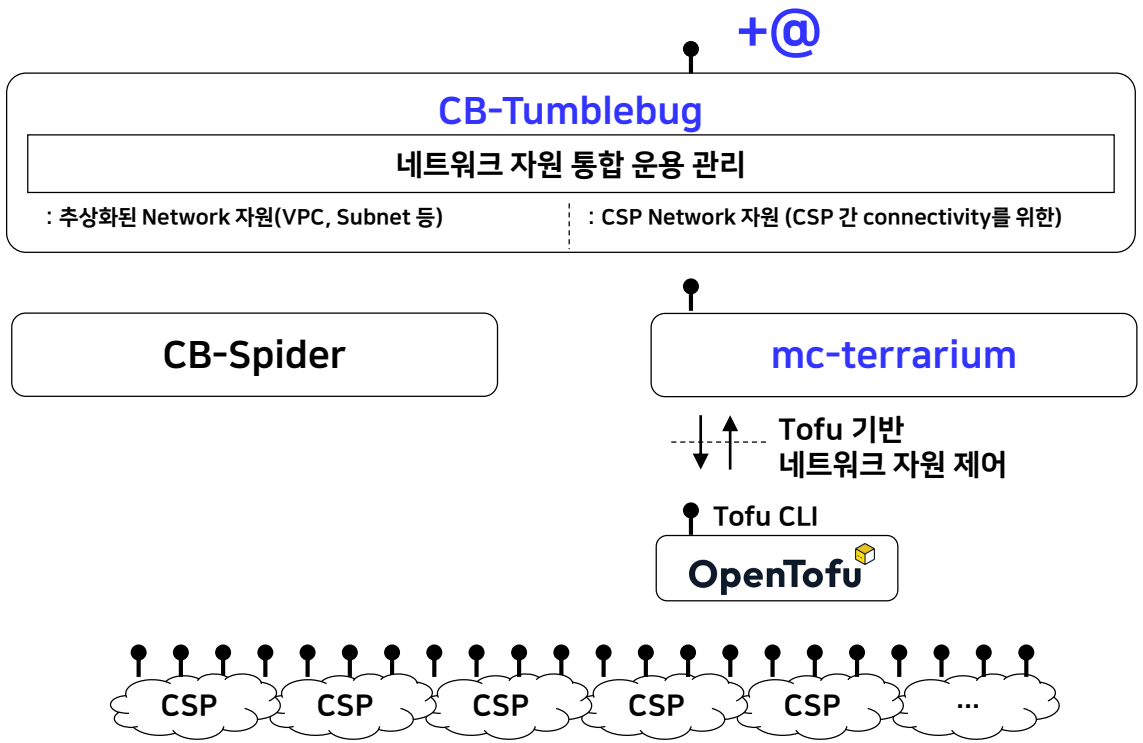
[GCP to AWS VPN tunnel]
- GCP: Region, VPC name
- AWS: Region, VPC ID, subnet ID (to import routing table) ← for propagation setting

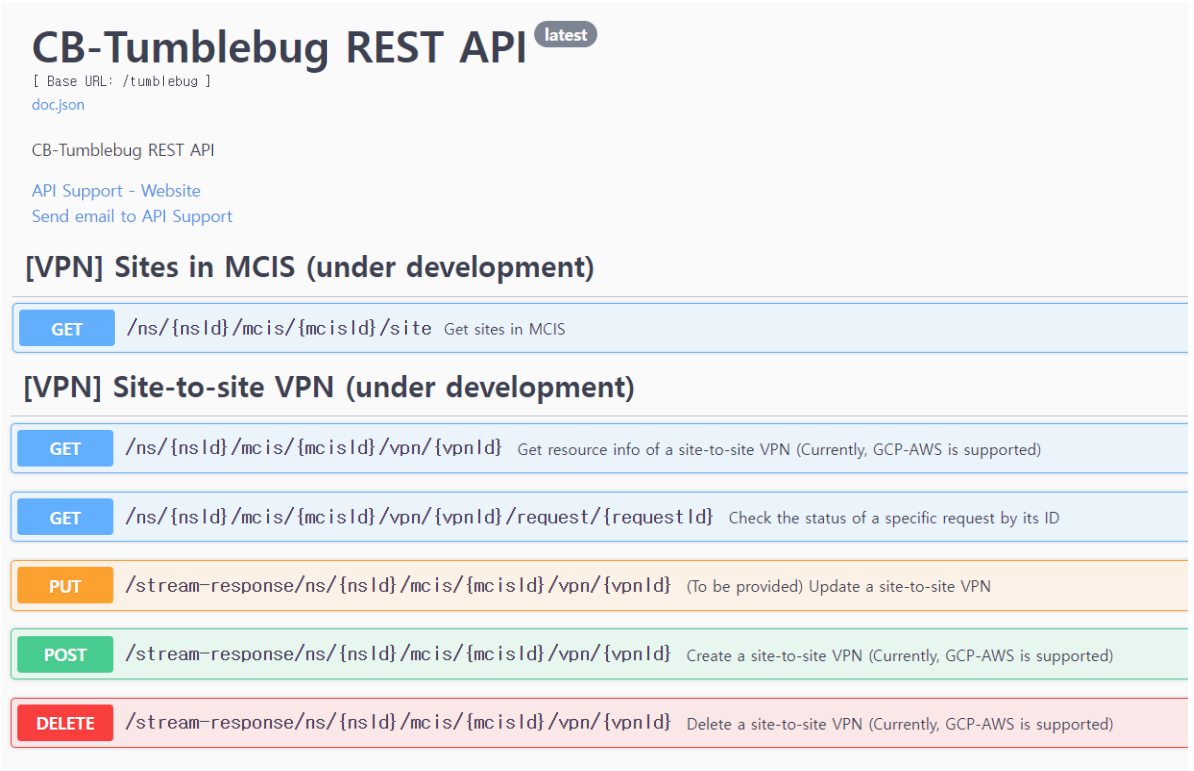
[GCP to Azure VPN tunnel]
- GCP: Region, VPC name
- Azure: Region, resource group name, virtual network name, GW subnet CIDR block

CB-Tumblebug에 Site-to-site VPN Feature 추가 및 개선 중

(GCP-AWS VPN 추가 및 테스트 완료)

- ✓ MCIS에 포함된 Sites 정보 제공
- ✓ Site-to-site VPN 생성, 수정, 조회, 삭제 제공





Site-to-site VPN 관련 APIs

Tumblebug APIs 요약 설명

Sites 정보 (from MCIS 정보)

GET /ns/{nsId}/mcis/{mcisId}/site Get sites in MCIS

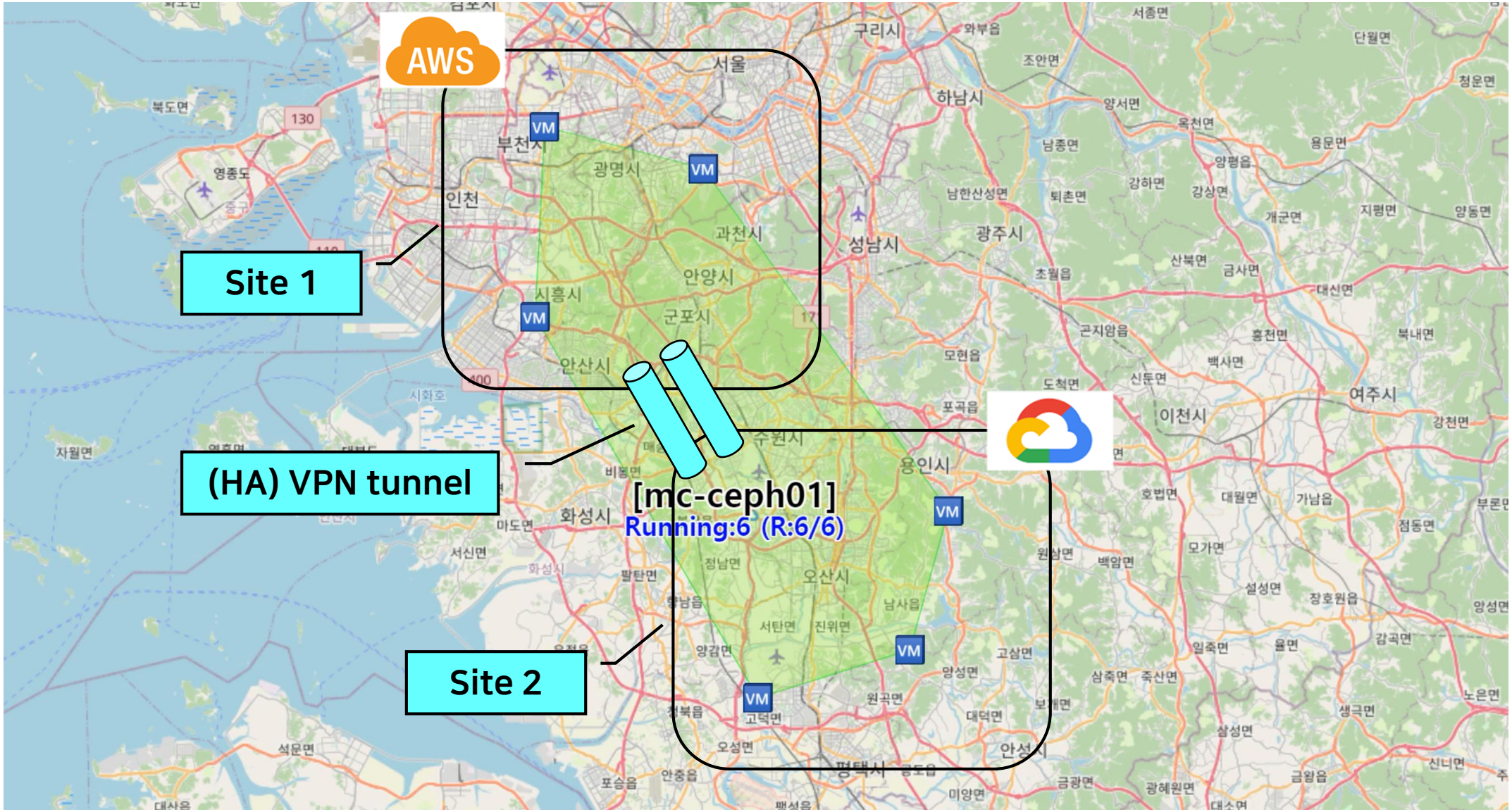

```
{
 "count": 3,
 "mcisId": "mcis-01",
 "nsId": "ns-01",
 "sites": {
  "aws": [
   {
    "csp": "aws",
    "gatewaySubnetCidr": "xxx.xxx.xxx.xxx/xx",
    "region": "ap-northeast-2",
    "resourceGroup": "rg-xxxxx",
    "subnet": "subnet-xxxxx",
    "vnet": "vpc-xxxxx",
    "zone": "ap-northeast-2a"
   }
  ],
  "azure": [
   {
    "csp": "aws",
    "gatewaySubnetCidr": "xxx.xxx.xxx.xxx/xx",
    "region": "ap-northeast-2",
    "resourceGroup": "rg-xxxxx",
    "subnet": "subnet-xxxxx",
    "vnet": "vpc-xxxxx",
    "zone": "ap-northeast-2a"
   }
  ],
  "gcp": [
   {
    "csp": "aws",
    "gatewaySubnetCidr": "xxx.xxx.xxx.xxx/xx",
    "region": "ap-northeast-2",
```


Site-to-site VPN

POST /stream-response/ns/{nsId}/mcis/{mcisId}/vpn/{vpnId} Create a site-to-site VPN (Currently, GCP-AWS


```
{
 "site1": {
  "csp": "aws",
  "gatewaySubnetCidr": "xxx.xxx.xxx.xxx/xx",
  "region": "ap-northeast-2",
  "resourceGroup": "rg-xxxxx",
  "subnet": "subnet-xxxxx",
  "vnet": "vpc-xxxxx",
  "zone": "ap-northeast-2a"
 },
 "site2": {
  "csp": "aws",
  "gatewaySubnetCidr": "xxx.xxx.xxx.xxx/xx",
  "region": "ap-northeast-2",
  "resourceGroup": "rg-xxxxx",
  "subnet": "subnet-xxxxx",
  "vnet": "vpc-xxxxx",
  "zone": "ap-northeast-2a"
 }
}
```

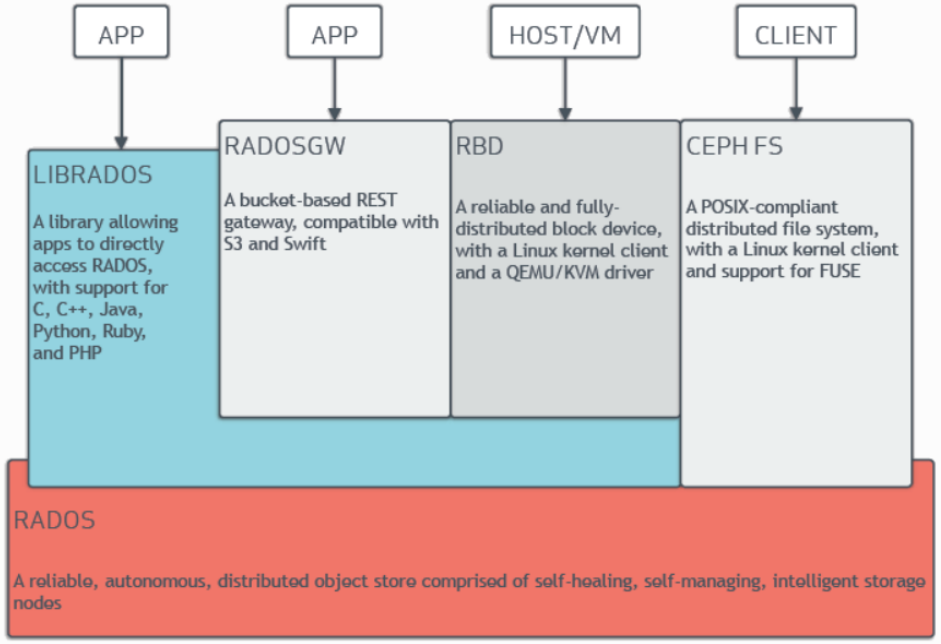
(결과/예시) MCIS에 Site-to-site VPN 보강



(시연 참고) Ceph

ARCHITECTURE

Ceph uniquely delivers **object, block, and file storage** in one unified system. Ceph is highly reliable, easy to manage, and free. The power of Ceph can transform your company's IT infrastructure and your ability to manage vast amounts of data. Ceph delivers extraordinary scalability–thousands of clients accessing petabytes to exabytes of data. A Ceph Node leverages commodity hardware and intelligent daemons, and a Ceph Storage Cluster accommodates large numbers of nodes, which communicate with each other to replicate and redistribute data dynamically.



RADOS: A Scalable, Reliable Storage Service for Petabyte-scale Storage Clusters

Sage A. Weil Andrew W. Leung Scott A. Brandt Carlos Maltzahn
University of California, Santa Cruz
{sage,aleung,scott,carlosm}@cs.ucsc.edu

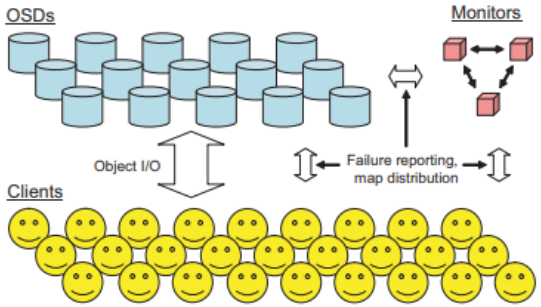
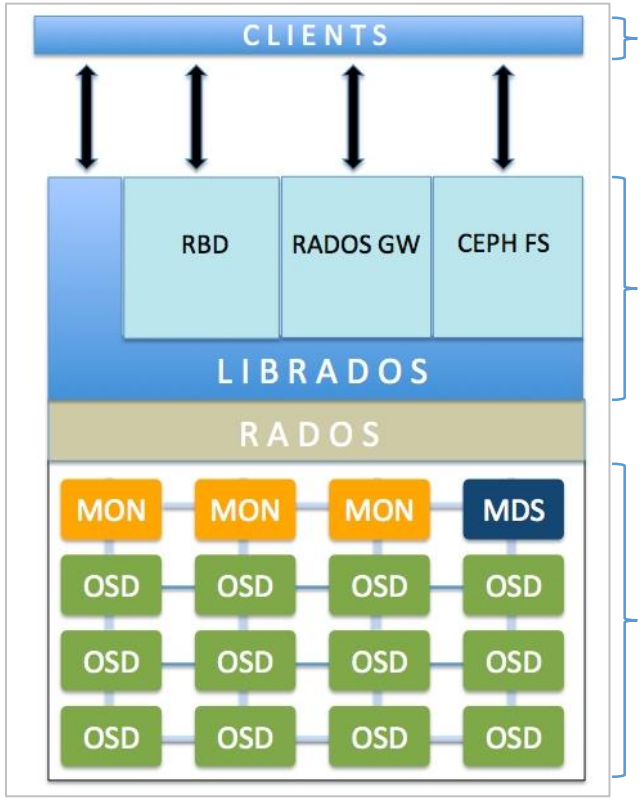


Figure 1: A cluster of many thousands of OSDs store all objects in the system. A small, tightly coupled cluster of monitors collectively manages the cluster map that specifies cluster membership and the distribution of data. Each client exposes a simple storage interface to applications.

*** OSDs: Object Storage Devices**

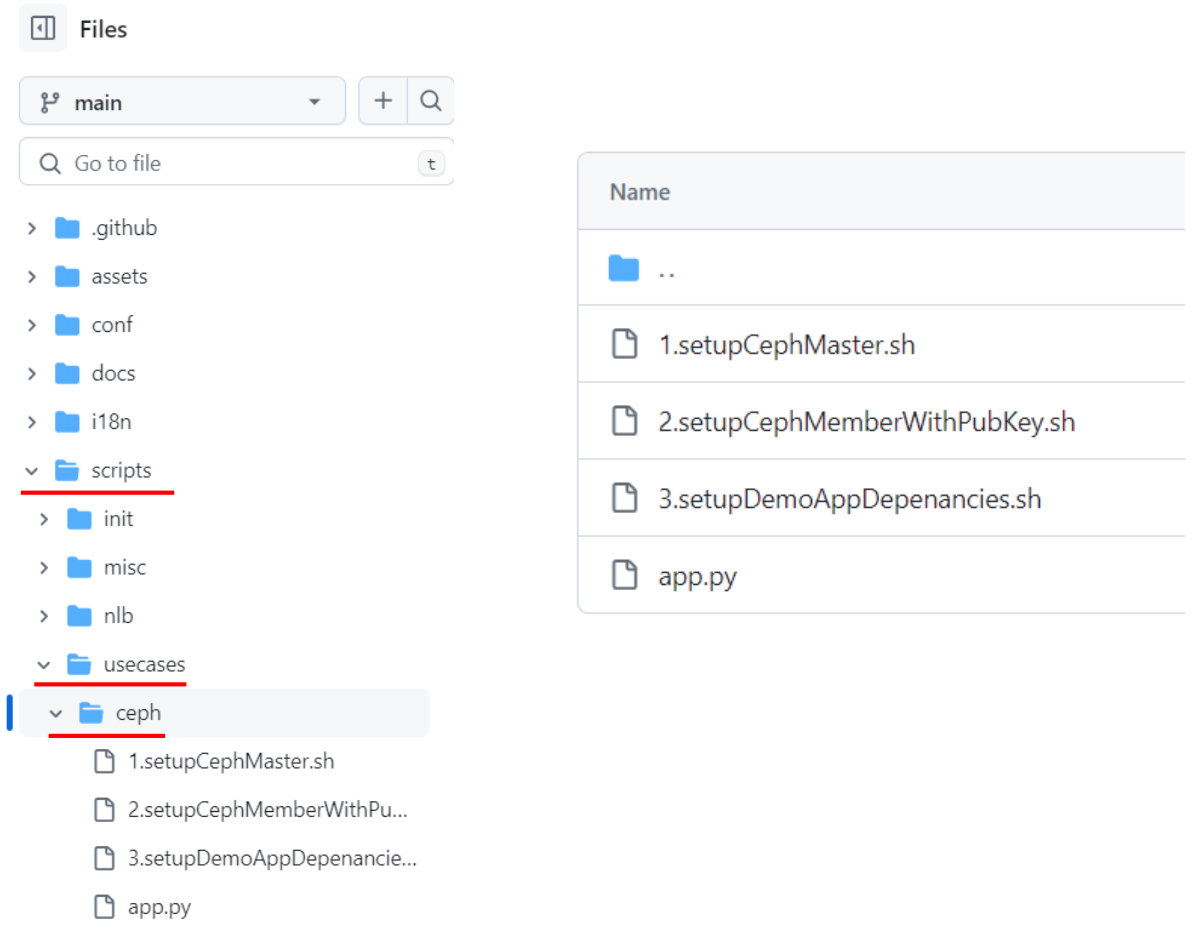


Ceph – the architectural overview

https://docs.ceph.com/en/quincy/architecture/
https://ceph.io/assets/pdfs/weil-rados-pdsw07.pdf
https://subscription.packtpub.com/book/cloud-and-networking/9781784393502/1/ch01lvl1sec12/ceph-the-architectural-overview

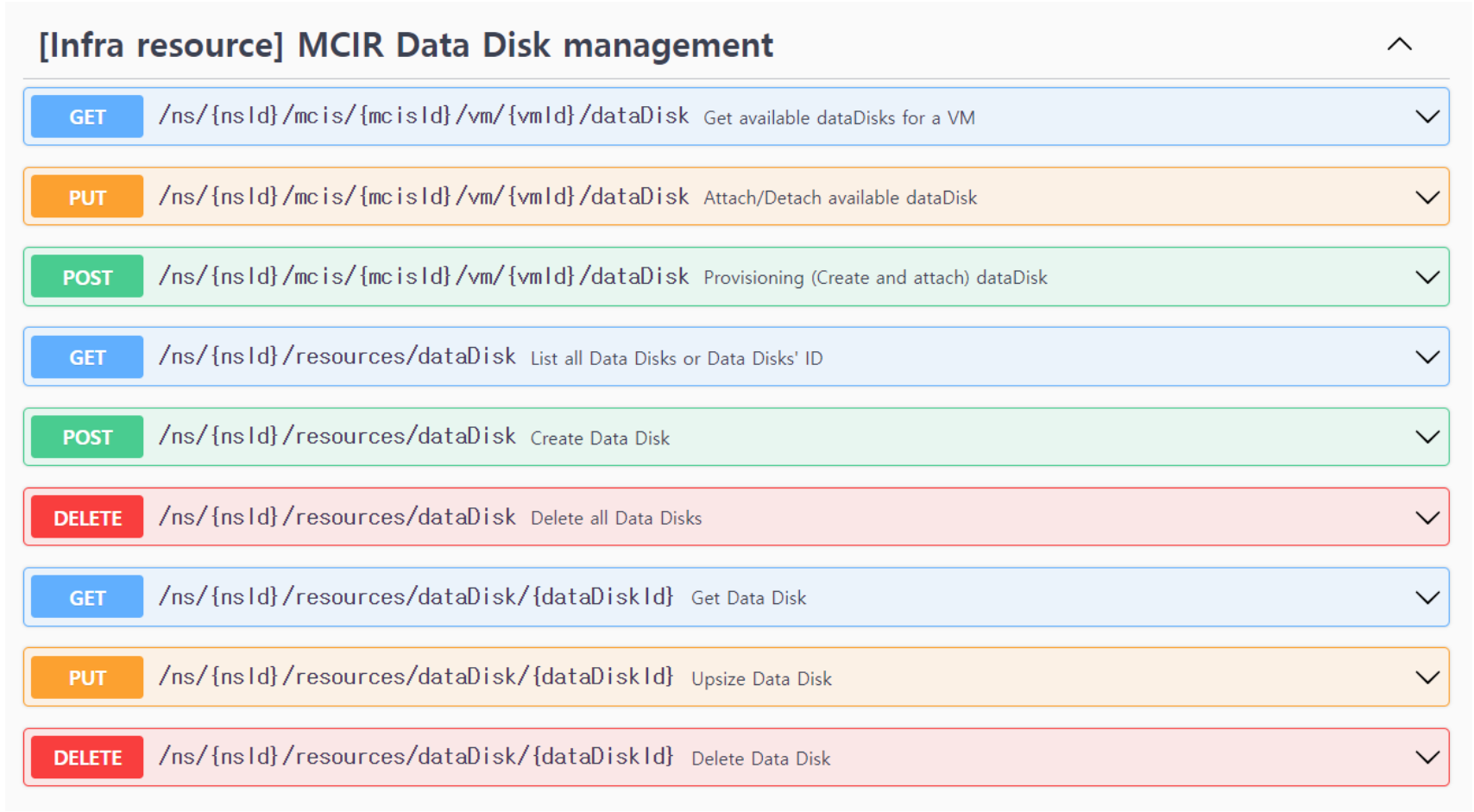
(시연 참고) Ceph 관리 및 멤버 노드 설치 스크립트, Demo App

CB-Tumblebug 저장소



https://github.com/cloud-barista/cb-tumblebug/tree/main/scripts/usecases/ceph

(시연 참고) 멀티 클라우드 인프라 블록 스토리지 생성 및 연동 API 형상



가용 DataDisk 리스트 조회

DataDisk VM 연동/분리

DataDisk 리스트 조회

DataDisk 생성

DataDisk 전체 삭제

DataDisk 조회

DataDisk 크기 변경

DataDisk 삭제

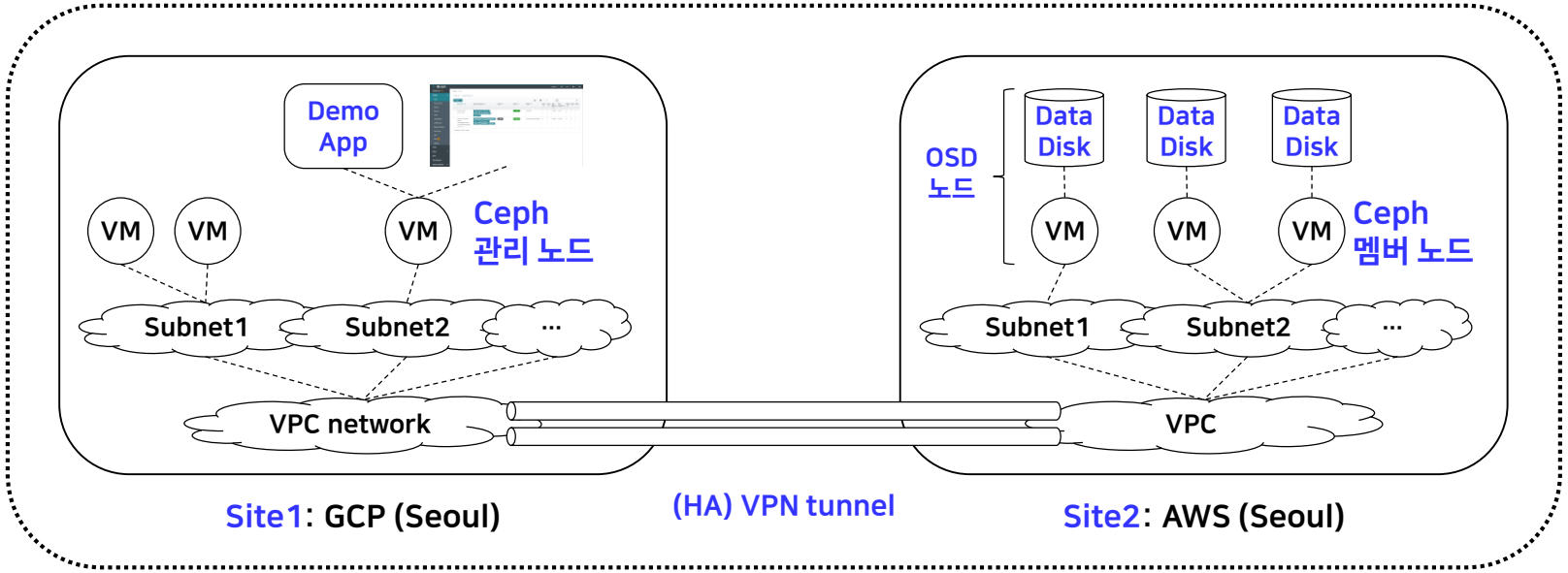
DataDisk 관리 API

(시연) 멀티 클라우드에서 VPN 기반의 Ceph 활용

시연 목적 및 요약

- (기술 시연) Spider 및 Tumblebug에서 생성한 멀티 클라우드 인프라 (MCI)에 Terrarium을 통한 Site-to-site VPN 보강/구축 시연
- (유스케이스 시연) VPN 기반 멀티 클라우드 인프라에서 Ceph 클러스터 구성 및 Data Disk 활용 시연

시연 개념도



* OSD: Object Storage Daemon

MCI: Multi-Cloud Infrastructure

(참고) AtoZ 영상, 멀티 클라우드에서 VPN 기반의 Ceph 활용

A-Z 시연 영상 경로: https://drive.google.com/drive/folders/1MZNhQMAeM3ON561O_FbuVJGljxovsK80?usp=sharing

A-Z 시연 영상 소개:

영상 #1 – 서브시스템 설정 및 구동

1. (Spider) 서버 설정 및 구동
2. (terrarium) 서버 설정 및 구동
3. (terrarium) API docs 접속
4. (Tumblebug) 서버 설정 및 구동
5. (Tumblebug) API docs 접속
6. (mapui) 설정 및 구동
7. (mapui) Website 접속

영상 #2 – MCIS 생성 및 확인

1. (Tumblebug) Namespace 생성
2. (mapui) MCIS 생성 요청
    1. VM spec 설정 (vcpu: 4, memory: 8GiB)
    2. Namespace 선택
    3. MCIS를 생성할 지역 선택 (Seoul AWS & GCP)
    4. 생성할 VM 수 입력
    5. MCIS 생성 요청
3. (mapui) MCIS 생성 확인

영상 #3 – VM 접속 및 연결성 테스트

1. (mapui) MCIS 접속 정보 및 SSH Key 획득
2. (Terminal) SSH key 저장 및 권한 설정
3. (Terminal) AWS의 VM 접속
4. (Terminal) GCP의 VM 접속
5. (Terminal) 상호 Ping test
6. (Terminal) 상호 Ping 불가함 확인

영상 #4 – Site-to-site VPN 보강

1. (사전 준비) Tumblebug 구동 현황 및 API Docs
2. (사전 준비) terrarium 구동 현황 및 API Docs
3. (Tumblebug) Sites 정보 조회
4. (Tumblebug) Site-to-site VPN 생성 요청
5. (terrarium) 연계·구동 현황 시연 (API 비동기 처리)
6. (terrarium) 환경 및 보강 자원 생성 시연
7. (AWS 및 GCP) Console에서 생성된 자원 확인

영상 #5 – VPN 기반 Ceph 클러스터 구축

1. (관리 노드) 설정/설치 (GCP Seoul 리전의 VM 1대)
2. (멤버 노드) 설정/설치 (AWS Seoul 리전의 VM 3대)
3. (관리 노드) 멤버 노드 연동
4. (Ceph Dashboard) 클러스터 구축 현황 시연
5. (Tumblebug) Data disk 생성 및 멤버 노드 연동
6. (Ceph Dashboard) OSDs 연동 현황 시연

영상 #6 – DemoApp을 통한 OSDs 활용 시연

1. (관리 노드) DemoApp 실행
2. (DemoApp) 웹사이트 접속
3. (DemoApp) Dummy data 생성 및 저장
4. (DemoApp) Dummy data 조회 및 출력
5. (Ceph Dashboard) OSDs I/O 변화 시연

멀티 클라우드에서 VPN 기반의 Ceph 활용 시연

Epilogue - Cloud-Barista가 OpenTofu를 만났을 때

스쳐 지나칠 수 있었던 Cloud-Barista와 OpenTofu의 인연은
유의미한 mc-terrarium 을 만들어 냈고, 멀티 클라우드 인프라를 보강할 수 있게 되었다 ^^;;;

1. 인프라 운영자가 다양한 형태의 인프라를 관리 할 수 있는 환경 제공
   - 예) Directory + Resource Group ID

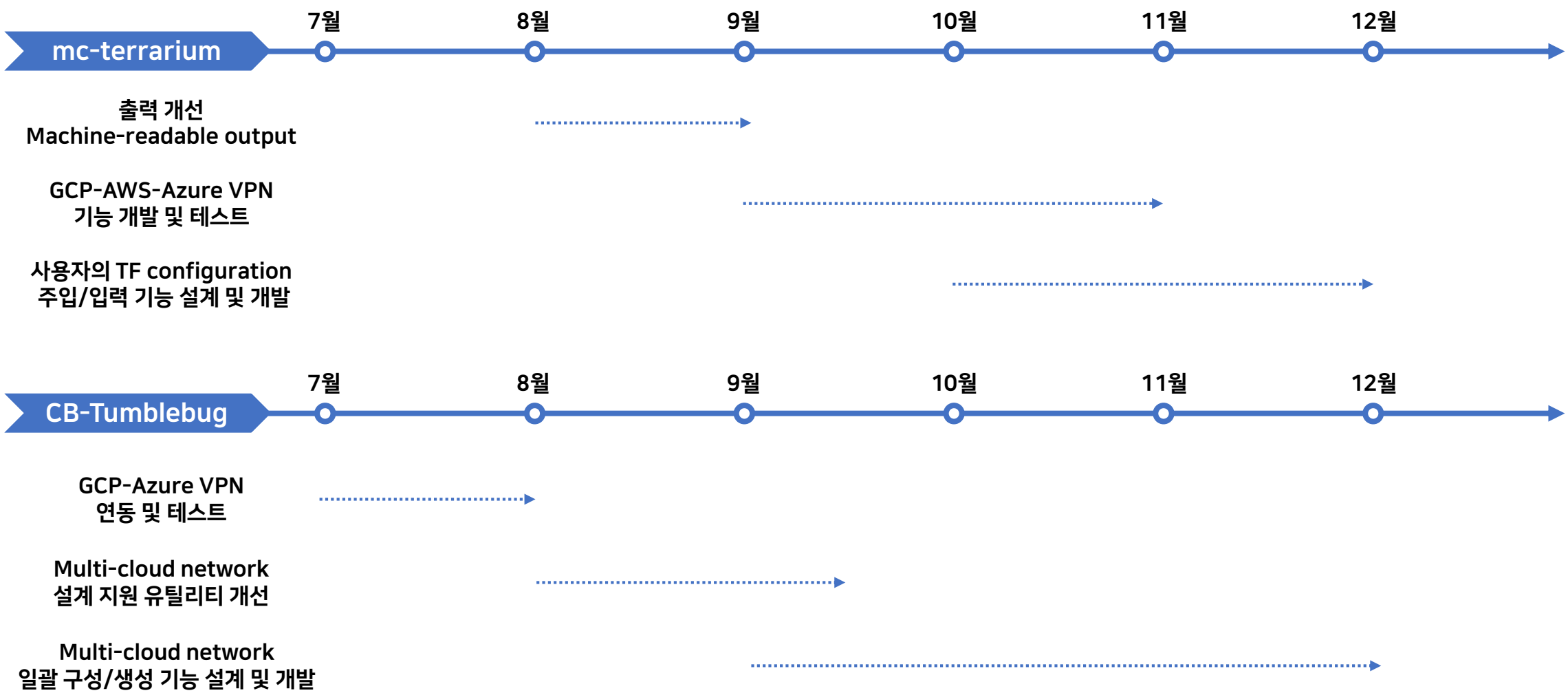
2. 정적인 TF configuration을 Template 화 ➔ 유연한 인프라 구축/구성 지원
   - 예) GCP to AWS VPN tunnel 구성을 위한 Template을 제공, 사용자가 두 Site의 정보를 입력하여 VPN tunnel 생성

3. REST API를 통해 Cloud-Barista 서브시스템과 연계 가능
   - 예) Tumblebug+Spider를 통한 MCIS 생성 후, 두 Site(예, VPCs)에 대한 정보를 활용하여 VPN tunnel 자원 보강/생성

4. API 요청에 대한 동기/비동기 처리 지원 (자생 생태계^^;;)
   - 예) Azure VPN gateway 생성에 약 30분 정도가 소요되는데…, 이를 poc-mc-net-tf 내부에서 지속 수행
   - ※ 상태 조회 API를 별도로 제공

⬆ 특장점 of mc-terrarium: multi-cloud terrarium ☺

향후 계획


7월
8월
9월
10월
11월
12월
mc-terrarium
출력 개선
Machine-readable output
GCP-AWS-Azure VPN
기능 개발 및 테스트
사용자의 TF configuration
주입/입력 기능 설계 및 개발
7월
8월
9월
10월
11월
12월
CB-Tumblebug
GCP-Azure VPN
연동 및 테스트
Multi-cloud network
설계 지원 유틸리티 개선
Multi-cloud network
일괄 구성/생성 기능 설계 및 개발


멀티 클라우드에 진심인 사람들의 이야기
멀티/분산 클라우드, 차세대 클라우드를 향한 도전과 기회
Cloud-Barista Community the 9th Conference
감사합니다.
https://github.com/cloud-barista
https://cloud-barista.github.io
김 윤 곤 / yunkon.kim@etri.re.kr
본 발표자료에는 네이버에서 제공한 나눔글꼴이 적용되어 있습니다.