

クラウドオペレーティングモデルは ネットワークの世界でどこまでできるか？

How can the cloud operating model be applied to current network operations?

Shogo Katsurada

Partner Solutions Engineer at HashiCorp

He/him

January 28, 2022

Shogo Katsurada / 桂田 祥吾

Partner Solutions Engineer at HashiCorp

2005年 4月 - Cisco Systems.

2021年 8月 - HashiCorp.

Janog との関わり

- Janog41.5
 - パブリッククラウドでの仮想ルーター利用と自動化
- Janog42
 - To disaggregate or not to disaggregate – そして良いディスアグリゲーション. 悪いディスアグリゲーションとは

Twitter: @shogokatsurada



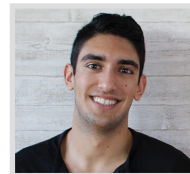
HashiCorpについて



Unlocking Cloud Operating Model

HashiCorpは、すべてのビジネスにおけるクラウドオペレーションモデルの制約を解放し、お客様のデジタル変革を成功に導きます。

HashiCorpの製品を利用することにより、アプリケーションとそれを支えるインフラストラクチャの「プロビジョニング」「セキュリティ保護」「実行」「接続と連携」が可能になります。



設立
2012

従業員
1,700+

資金調達
\$340M

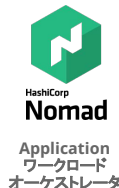
2021/12/9 IPO

市場評価額
\$5B +

直近の受賞:

- ・Forbes Cloud 100の4位に3年連続ランクイン
- ・Enterprise Tech 30にて2年連続 Late Stage部門で1位

キー製品



Enterprise版+OSS版

OSS版

ハイブリッドなインフラへのシフト



Traditional Datacenter

"Static"



Dedicated
Infrastructure



Private
Cloud

Modern Datacenter

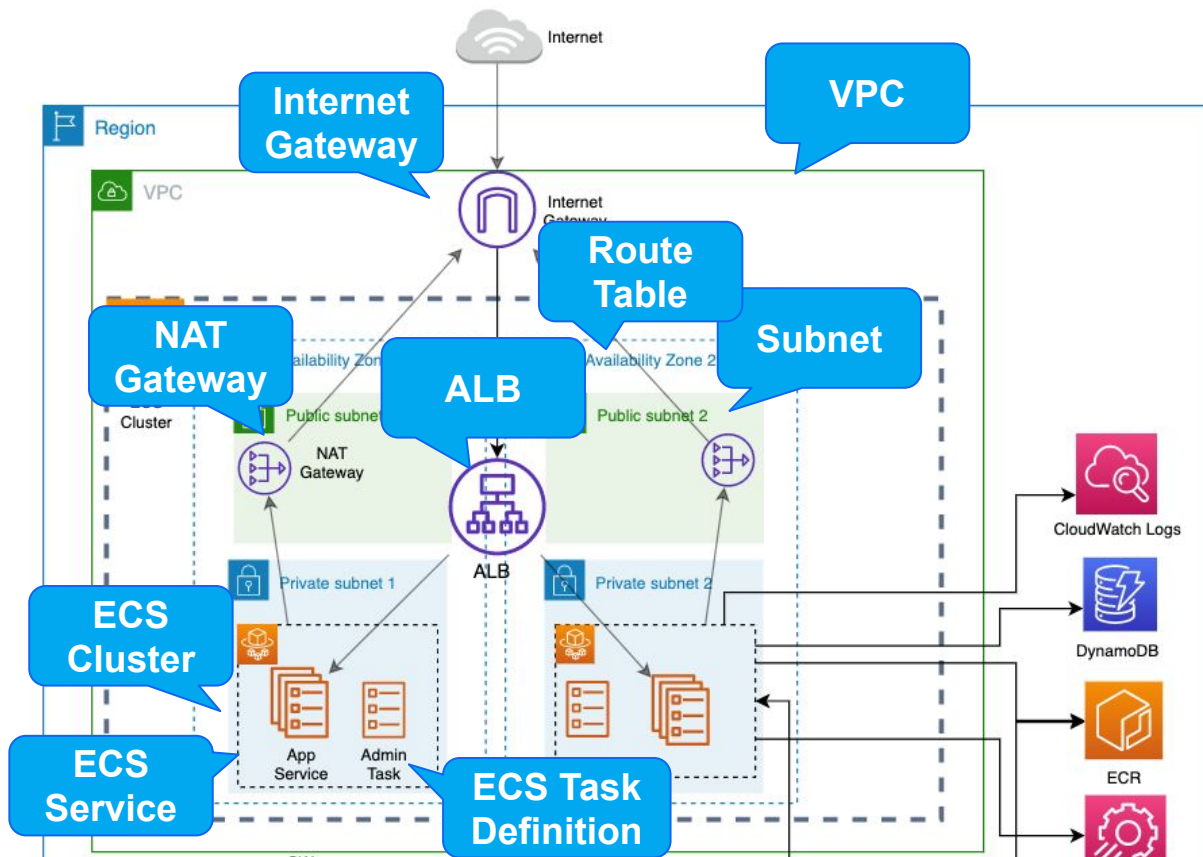
"Dynamic"



AWS + Azure + GCP + ...

	Run Development	専用 プラットフォーム	→	様々なプラットフォームへの自動デプロイ
	Connect Networking	ホストベース	→	サービスベース
	Secure Security	高信頼性 IP-based	→	低信頼性 Identity-based
	Provision Operations	専用マシン	→	オンデマンド

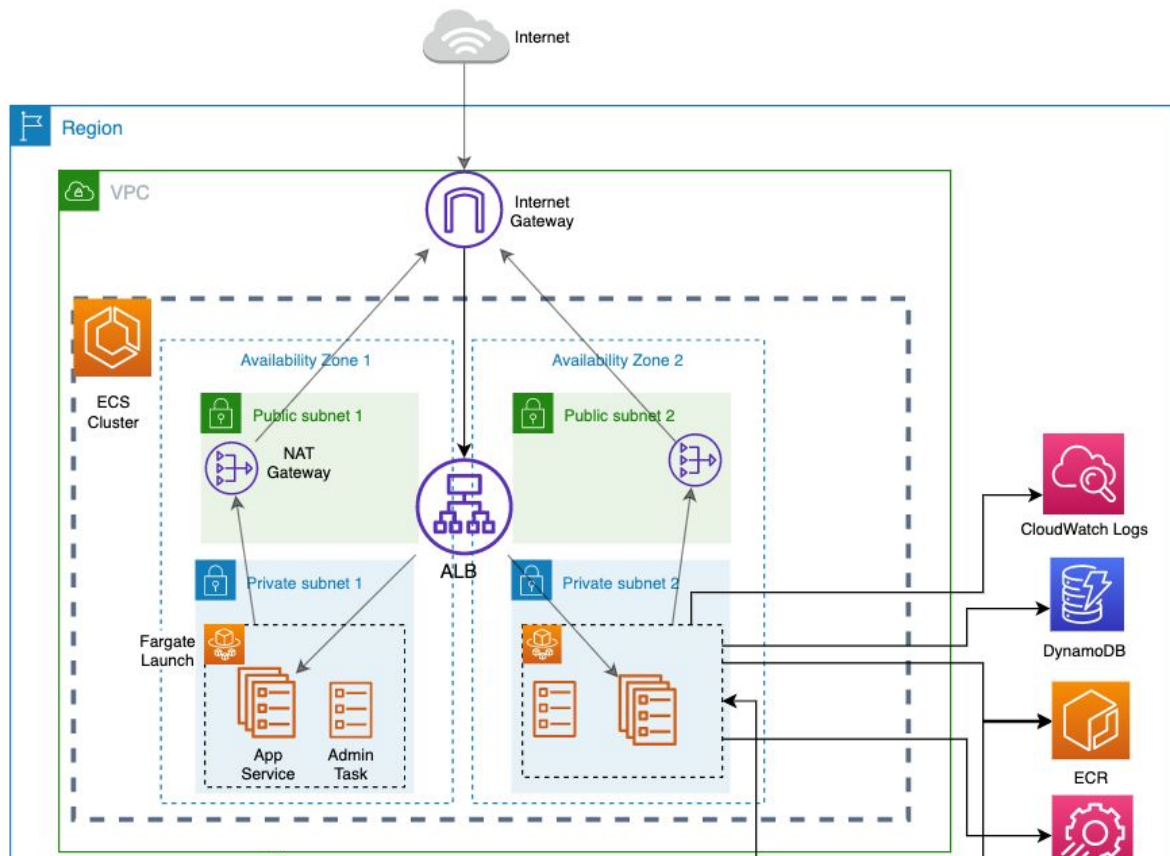
クラウド上で動くアプリの構成例



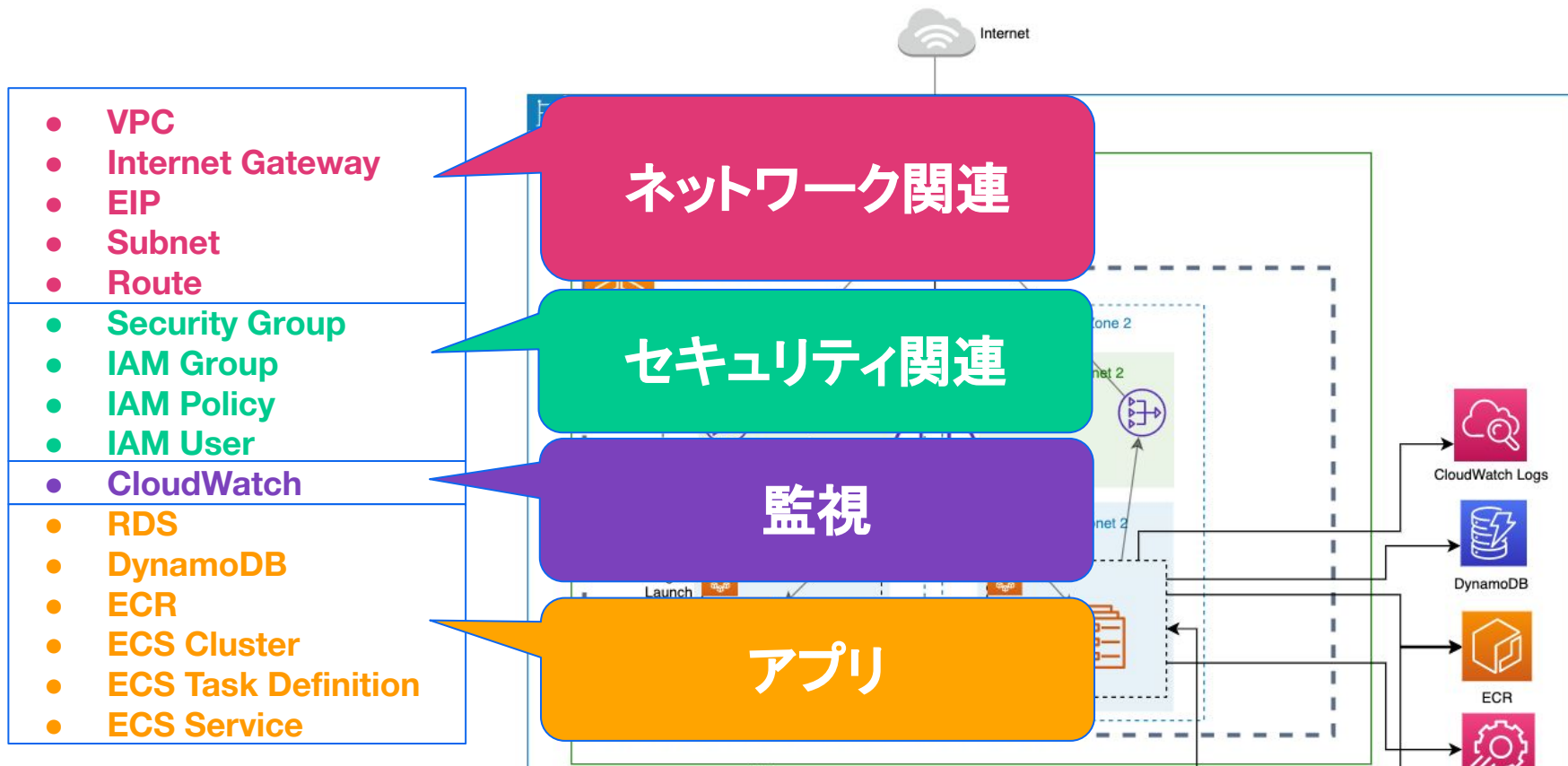
クラウド上で動くアプリの構成例



- VPC
- Internet Gateway
- EIP
- Subnet
- Route
- Security Group
- IAM Group
- IAM Policy
- IAM User
- CloudWatch
- RDS
- DynamoDB
- ECS Cluster
- ECS Task Definition
- ECS Service



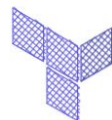
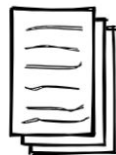
クラウド上で動くアプリの構成例



クラウド上で動くアプリの構成例



<ul style="list-style-type: none">● VPC● Internet Gateway● EIP● Subnet● Route
<ul style="list-style-type: none">● Security Group● IAM Group● IAM Policy● IAM User
<ul style="list-style-type: none">● CloudWatch
<ul style="list-style-type: none">● RDS● DynamoDB● ECR● ECS Cluster● ECS Task Definition● ECS Service



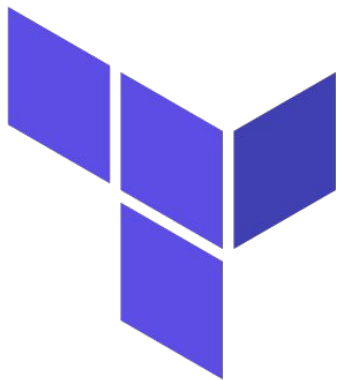
クラウドの自動化では、

- ネットワーク
- セキュリティ
- 監視設定
- アプリケーション

ほとんどのコンポーネントが、一貫したワークフローで運用されている (DevOps)

Terraform はマルチクラウドに対応したオープンソースの IaC のツールとしてデファクトスタンダードな存在

Terraform is 何?



HashiCorp

Terraform

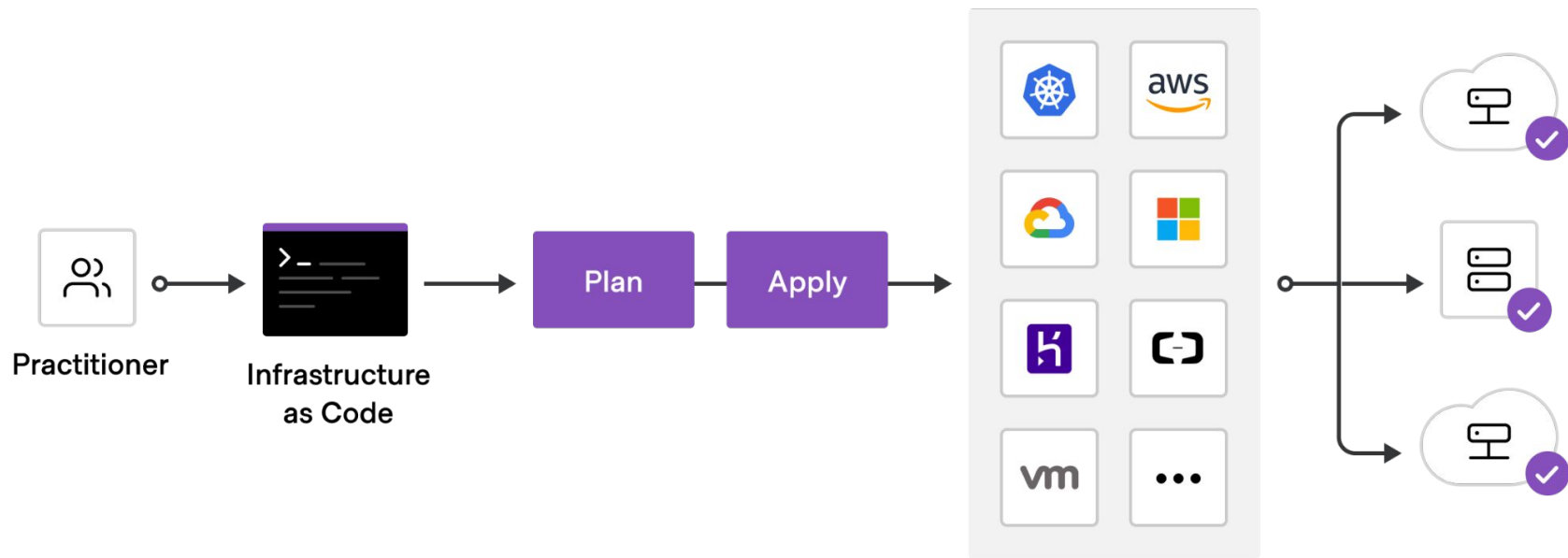
Terraform はオープンソースのプロビジョニングツール
(2014年～、2021/6 に version 1.0 GA, 現在latest 1.1.4)

Go で書かれたシングルバイナリ。Terraform はLinux、
Windows、MacOS などクロスプラットフォームで動作。

Terraform では

- プロビジョニング先を **provider**、
- 設定するインフラを **resource** として扱い
- **宣言的に**プロビジョニングを実行します
- また **state** ファイルにてライフサイクルの管理を行います(ドリフト検出が可能)

様々なインフラへのプロビジョニング



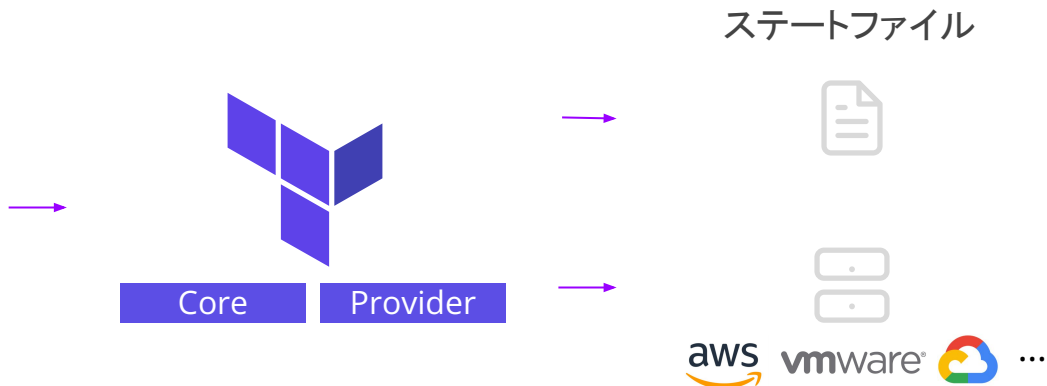
- 150 を超える公式プロバイダ
- 1,300 を超えるCommunity プロバイダ

Terraform によるプロビジョニングイメージ



HashiCorp Configuration Language

```
resource "vsphere_virtual_machine" "vm" {  
  name          = "terraform-test"  
  resource_pool_id = "*****"  
  datastore_id  = "*****"  
  num_cpus     = 2  
  memory       = 1024  
  
  network_interface {  
    network_id = "*****"  
  }  
  
  disk {  
    label = "disk0"  
    size  = 20  
  }  
}
```



実行可能なパラメータシート
(のようなもの)

Terraformによるプロビジョニングイメージ



```
resource "aws_instance" "web_ec2" {
  ami    = var.ami
  count  = 1
  tags = merge(var.tags, map(
    "Name", "instance-a",
    "ENV", "DEV"
  ))
  instance_type = "t2-small"
  vpc_security_group_ids =
"my-sec-group"
  associate_public_ip_address = true
}
```



```
resource "aws_vpc" "main" {
  cidr_block      = "10.0.0.0/16"
  instance_tenancy = "default"

  tags = {
    Name = "main"
  }
}
```

```
resource "aws_subnet" "main" {
  vpc_id      = aws_vpc.main.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "Main"
  }
}
```

『あるべき形』をコードとして記述



Terraform のプロバイダが増加中

Cisco, Citrix, and Fortinet Among New Verified Terraform Providers

Cloud, SaaS だけでなく
対応ネットワークベンダーも増加中

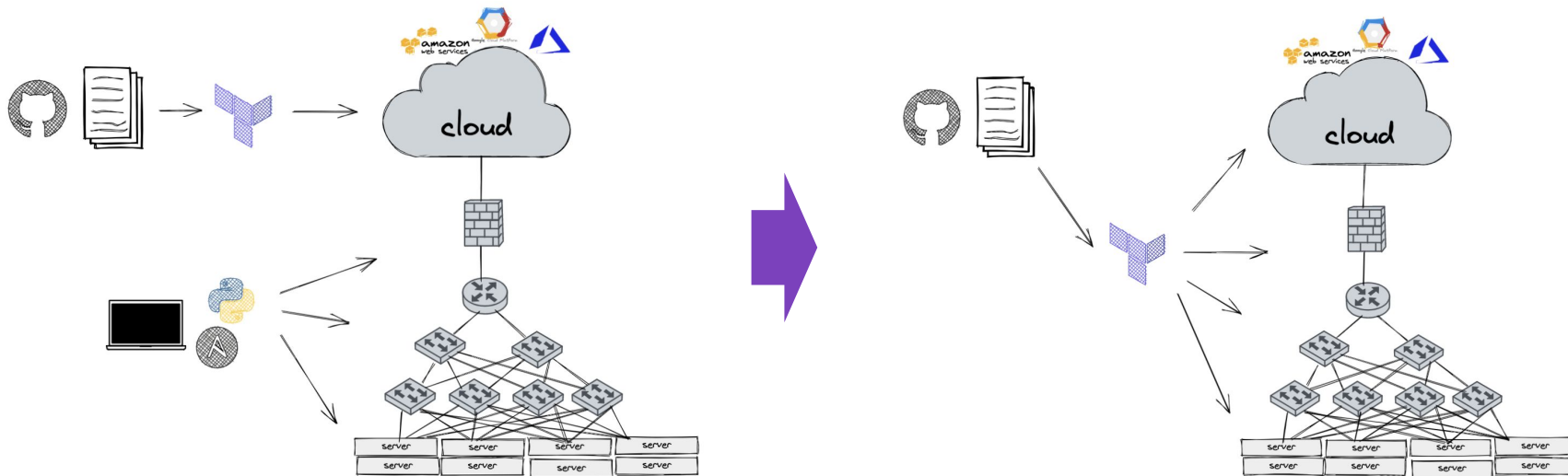


- Cisco
- Citrix
- Fortinet
- Purestorage
- F5
- Arista
- VMware
- Paloalto
- A10
- Infoblox
- Bluecat
- Cloudflare
- Fastly



てことは...

クラウド運用と同様の 運用パイプラインに組み込むこともできそう？



registry.terraform.io でプロバイダを検索



Providers

Search Providers and Modules

Filters:

- Tier: Official, Verified, Community
- Category: HashiCorp Platform, Public Cloud, Asset Management, Cloud Automation, Communication & Messaging, Container Orchestration, Continuous Integration/Deployment (CI/CD), Data Management, Database, Infrastructure (IaaS), Logging & Monitoring, Networking, Platform (PaaS), Security & Authentication, Utility

Providers listed:

- cloudflare (by: cloudflare)
- nsxt (by: vmware)
- dnsmple (by: dnsmple)
- aviatrrix (by: AviatrrixSystems)
- fortios (by: fortimdev)
- panos (by: PaloAltoNetworks)
- aci (by: CiscoDevNet)
- infoblox (by: infobloxopen)
- checkpoint (by: CheckPointSW)
- mso (by: CiscoDevNet)
- fortimanager (by: fortimdev)
- zerotier (by: zerotier)
- alkira (by: alkiranet)
- citrixadc (by: citrix)
- avi (by: vmware)
- cloudeos (by: aristanetworks)
- dcnm (by: CiscoDevNet)
- ngrok (by: ngrok)
- bluecat (by: bluecatalabs)

<https://registry.terraform.io/search/providers?category=networking>

Providers

Search Providers and Modules

Filters:

- Tier: Official, Verified, Community
- Category: HashiCorp Platform, Public Cloud, Asset Management, Cloud Automation, Communication & Messaging, Container Orchestration, Continuous Integration/Deployment (CI/CD), Data Management, Database, Infrastructure (IaaS), Logging & Monitoring, Networking, Platform (PaaS), Security & Authentication, Utility

Providers listed:

- Azure Active Directory (by: hashicorp)
- okta (by: okta)
- lacework (by: lacework)
- sdm (by: strongdm)
- venafi (by: Venafi)
- Active Directory (by: hashicorp)
- oktaasa (by: oktadeveloper)
- prismacloud (by: PaloAltoNetworks)
- dome9 (by: dome9)
- sysdig (by: sysdiglabs)
- boundary (by: hashicorp)
- onelogin (by: onelogin)
- onepassword (by: 1Password)
- vaultix (by: vaultix-security)
- sigsci (by: signalsciences)
- aquasec (by: aquasecurity)
- ciscoasa (by: CiscoDevNet)
- brtve (by: brtve)
- satori (by: SatoriCyber)
- fmc (by: CiscoDevNet)
- tetration (by: CiscoDevNet)

<https://registry.terraform.io/search/providers?category=security-authentication>

プロバイダを開発するための SDKもあるので、プロバイダを自作することも可能

対応リソースを確認する



Terraform では設定対象をリソースとみなす

- 追加したいリソースをコードに書く(宣言的)

```
resource "fortios_router_static" "test" {
  blackhole      = "disable"
  device         = "port1"
  distance      = 10
  dst            = "120.2.2.122/32"
  dynamic_gateway = "enable"
  status         = "enable"
  weight        = 2
}
```

- リソースとして定義されていないものは設定できない
 - 1行毎にコマンドを入力という設定はない
 - REST API を叩くリソースを提供しているプロバイダも存在

The screenshot shows the Terraform Registry page for the 'aci' provider. The page is titled 'aci' and 'Access Policies'. It displays a list of resources under 'Access Policies' and 'Resources'. The 'aci_cdp_interface_policy' resource is highlighted with a blue dot. The 'Example Usage' section shows a Terraform resource block for 'aci_cdp_interface_policy' with attributes like 'name', 'admin_st', 'annotation', 'name_alias', and 'description'. The 'Argument Reference' section lists the arguments for the resource.

NW関連 Terraform プロバイダの対応状況



2022年1月14日 現在での傾向

- プロバイダ数
 - Networking カテゴリ: 103
 - Security & Authentication カテゴリ: 155
- 幾つかの主要ベンダーでの対応が進んでいる
- NW製品は、クラウド親和性の高い製品、SDN製品が多い
 - APIが叩ける製品, 仮想アプライアンスなど
- FW, LB の対応は、そこそこ進んでいる印象
 - ACL/ポリシーの追加/削除など、宣言型の需要は多そう
- ほとんどがここ1-2年以内リリースされたもの
 - 頻繁にアップデートされている

The image features a dark blue background with decorative geometric patterns. In the top-left corner, there are two overlapping squares: one with a grid of small white dots and another with diagonal white lines. In the bottom-right corner, there is a large square with a grid of small white dots.

Demo

ドリフト検出の例

Edit Static Route

Dynamic Gateway

Destination **Subnet** Internet Service
131|2.2.122/255.255.255.255

Interface public (port1)

Gateway Address **Dynamic** Specify 10.1.0.1

Administrative Distance 10

Comments Write a comment... 0/255

Status **Enabled** Disabled



shogokatsurada1 triggered a run from UI a few seconds ago Run Details

Plan finished an hour ago Resources: 0 to add, 1 to change, 0 to destroy

Started a few seconds ago > Finished a few seconds ago

Agent pool no-credential-demo Agent ECS_Fargate

~ 1 to change

Filter resources by address.. Terraform 1.1.3 Download raw log

~ fortios_router_static.test1

~ dst : "131.2.2.122/32" → "130.2.2.122/32"

~ id : "3"

... 19 unchanged attributes hidden

Download Sentinel mocks Sentinel mocks can be used for testing your Sentinel policies

Terraform で設定したアドレスを手動で変えてしまった！

Terraform plan を実行すると、差分が表示



まとめと議論に向けて

クラウドオペレーティングモデルで、運用の壁を "乗り越える"

Terraform のネットワークソリューション対応プロバイダも増えて来ました

- クラウド基盤、ネットワーク基盤も併せた自動化が捗りそう
- 宣言型、ドリフト検出など、手続き型とは違うアプローチの自動化
- 若手の育成、クラウド人材の採用に良いかも？

ただ、課題も多そう...

- 組織の問題、Cloud CoE/自動化チームがあるか？
- 対応製品の数(時間が解決しそうではある)



皆さんに聞いてみたいこと

- クラウド運用チームとネットワーク運用チームの壁
 - クラウド運用とネットワーク運用の壁“乗り越える”べき？
- Terraformをネットワークに適用するアプローチはいかがですか？
- Terraformをネットワークで使われてる方いらっしゃいますか？
- こういう事が出来たら使ってみたい

など、その他ご質問

Terraform ネットワークインフラ用のプロバイダは、数が増えてきましたが、まだまだリリースから日も浅いものが多いので、みなさんと一緒に知見をためていけると嬉しいです！



ハンズオン 企画しました

Terraformについて、勉強してみたい方、ぜひご参加ください～

<https://events.hashicorp.com/workshops/terraform-2/16/JANOG>

The screenshot shows a promotional banner for a HashiCorp workshop. On the left, the HashiCorp logo is at the top. Below it, the title 'Terraform Cloud on AWS - クラウドコストを管理する' is displayed in white text. Underneath the title, the date '2月16日 (水)' and time '1pm - 5pm JST | Terraform Cloud on AWS ハンズオンワークショップ' are listed. A blue button with the text '今すぐ登録する' is positioned below the time. At the bottom left of the banner, there is a link 'View All Workshops'. On the right side of the banner, there is a 3D isometric graphic of a purple cube with a white HashiCorp logo on its top face. The cube is resting on a dark grey base that features the logos of Microsoft Azure, Google Cloud, and Amazon AWS.

その他イベント <https://events.hashicorp.com/jpworkshops>



Thank You

shogo.katsurada@hashicorp.com

www.hashicorp.com

宣言型 vs 手続き型



Terraform (宣言モデル)

```
resource "aws_vpc" "default" {
  cidr_block      = var.vpc_cidr_block
  enable_dns_hostnames = true

  tags = {
    Name = "dev"
  }
}

resource "aws_subnet" "default" {
  availability_zone = var.availability_zone_a
  cidr_block        = "10.0.0.0/24"
  map_public_ip_on_launch = false
  vpc_id            = aws_vpc.default.id

  tags = {
    Name = "dev"
  }
}
```

7

Ansible (命令実行モデル)

```
tasks:
  - name: "VPC"
    ec2_vpc_net:
      name: "vpc"
      cidr_block: 10.0.0.0/24
      dns_hostnames: yes
      register: _vpc
  - name: "Public subnet"
    ec2_vpc_subnet:
      vpc_id: "{{ _vpc.vpc.id }}"
      az: "{{ item.az }}"
      cidr: "{{ item.cidr }}"
      resource_tags:
        Name: "dev"
      with_items: "{{ subnet }}"
```

Ansibleは、リソースの作成順序をユーザが定義するため、依存関係を考慮してコードを定義する必要がある。

他方Terraformは、リソースの依存関係を自動で解決するため、作成順序をコードで定義する必要がない。図のAnsibleで、VPCよりサブネットを先に記述した場合には、実行時にエラーとなる。

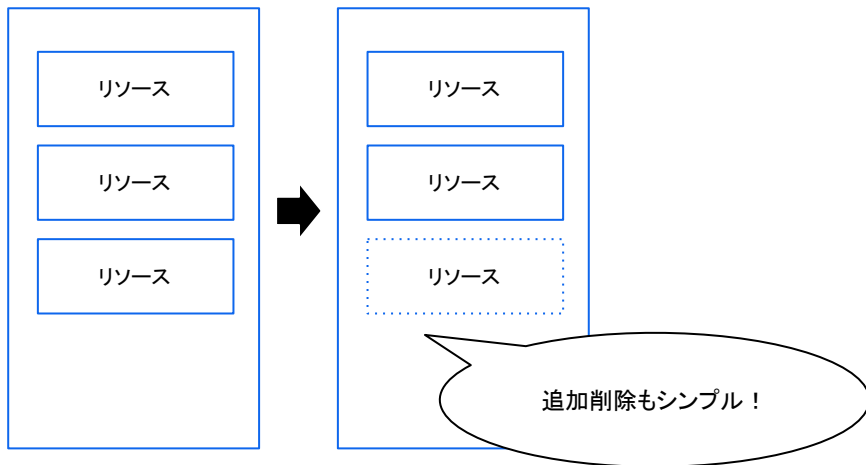
宣言型 vs 手続き型 - Cont.



Network自動化に関連して

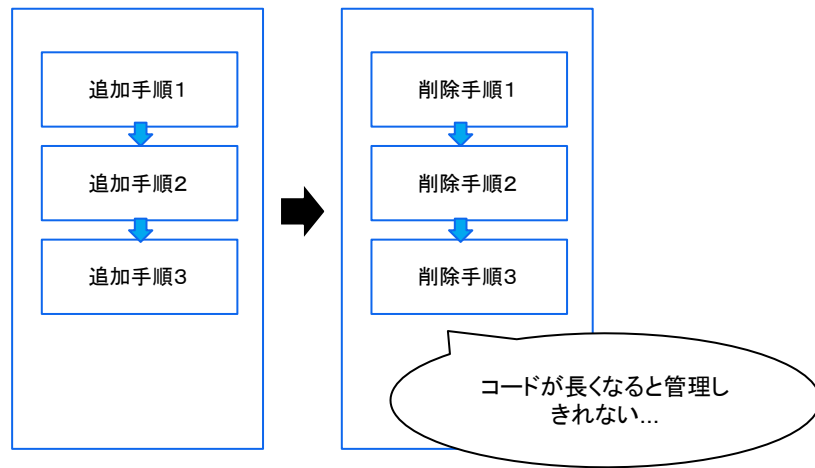
アクセスリストや、テナントを追加・削除するケース

宣言型の設定ファイル(状態を記述)



宣言型の場合は、あるべき状態を記載すればインフラに反映されるので、コードから消せば、削除される (VCS一つで状態の管理がしやすい)

手続き型の設定ファイル(手順を記述)



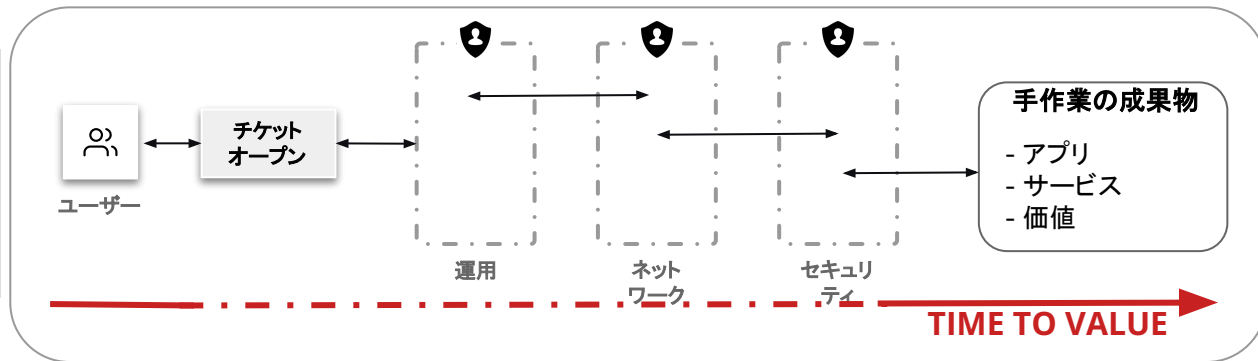
手続き型の場合は、手順書をコード化しているため、現在の状態は管理しない(別途管理が必要) 削除のためのコードを、順序通り書く必要があり、管理が複雑化しやすい

クラウドの世界でのワークフローは？

従来のワークフロー:

運用者がインフラに直接設定投入する

- Manufacturing Snowflakes
- Mutable インフラ
- 変更時間に時間とコストがかかる



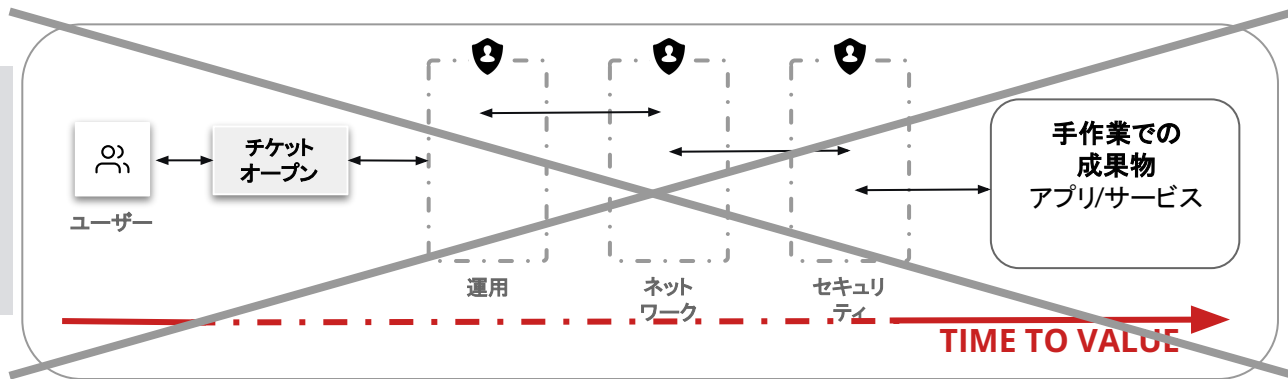
手動で行う時代には理にかなっていたが...
こういったワークフローは、
今日のビジネス要求を満たさないケースも...

クラウドの世界でのワークフローは？

従来のワークフロー:

運用者がインフラに直接設定投入する

- Manufacturing Snowflakes
- Mutable インフラ
- 変更時間に時間とコストがかかる



クラウド時代のワークフロー:

運用者が、IaCパイプラインの権限を持つ

- データドリブン -> IaC -> 自動化
- Immutable インフラ
- 変更は反復的、迅速かつ容易に

